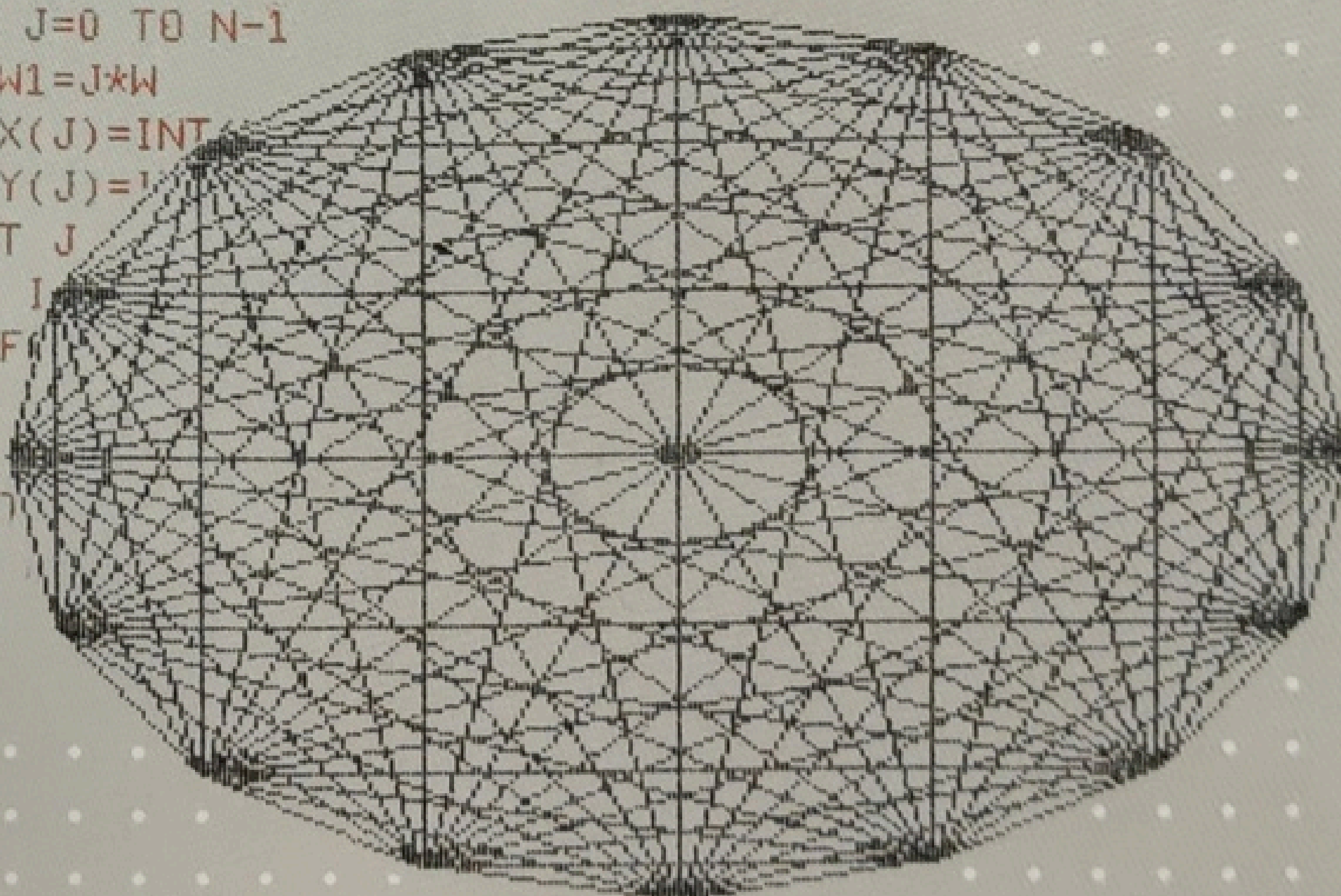
40 Grafische programma's voor de COMMODORE 64

Leer programmeren met
hoge resolutie graphics
in BASIC

Marcel Sutter

```
100 REM PROGRAMMA 5 DIAGONALEN IN EEN N-HOEK
110 PRINT CHR$(147)
120 INPUT "HALVE GROTE AS A<=160"; A
130 INPUT "HALVE KLEINE AS B<=100"; B
140 INPUT "HOEVEEL HOEKPUNTEN N<25"; N
150 PI=3.14159
160 HIRES 0,1
170 DIM X(N),Y(N)
180 U=160:V=100:H=0.5:W=(360/N)*PI/180
190 FOR J=0 TO N-1
200 :  W1=J*W
210 :  X(J)=INT
220 :  Y(J)=
230 NEXT J
240 FOR I
250 :  F
260 :
270 :
280 NEXT
290 GET
300 END
```



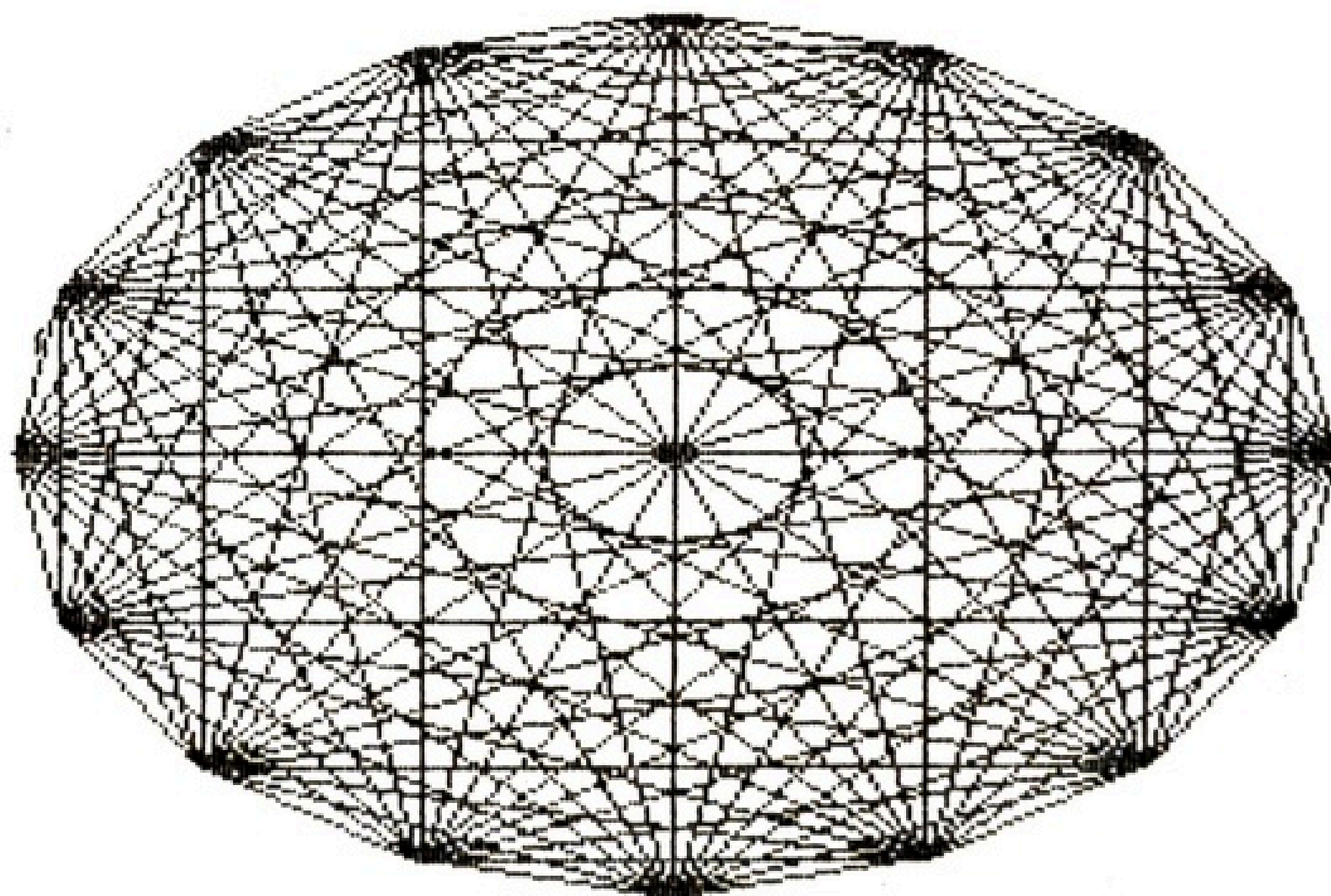
ACADEMIC SERVICE

40 GRAFISCHE PROGRAMMA's
voor de Commodore 64

40 Grafische programma's voor de COMMODORE 64

Leer programmeren met
hoge resolutie graphics
in BASIC

Marcel Sutter



ACADEMIC SERVICE

Oorspronkelijke titel: *Programmieren mit hochauflösender Grafik*
Verschenen bij: Mikro+Kleincomputer Informa Verlag AG, Luzern

Vertaling: Nok van Veen

CIP-GEGEVENS KONINKLIJKE BIBLIOTHEEK, DEN HAAG

Sutter, Marcel

40 grafische programma's voor de Commodore 64 / Marcel Sutter ; [vert. uit het Duits door Nok van Veen]. - Den Haag : Academic Service. - Ill.
Vert. van: Programmieren mit hochauflösender Grafik. - Luzern : Mikro + Kleincomputer Informa Verlag, 1984.
ISBN 90-6233-149-1
SISO 365.2 UDC 681.3.06
Trefw.: computerprogramma's.

Uitgegeven door: Academic Service
Postbus 96996
2509 JJ Den Haag
Zetwerk: multitASk, Blaricum
Druk: Krips Repro Meppel
Bindwerk: Meeuwis, Amsterdam
Omslagontwerp: Leo Bolt
ISBN 90 6233 149 1

Voorwoord bij de Nederlandse uitgave

De auteur, Marcel Sutter, heeft dit boek oorspronkelijk geschreven in standaard microsoft-BASIC. In zijn boek heeft hij in een aantal appendices de programma's opgenomen zoals ze voor de Commodore 64, de Apple II, de VIC-20 en de Sharp PC-1500 ingetikt moeten worden. Wij hebben besloten dit boek voor een aantal microcomputers afzonderlijk uit te geven. Het boek dat nu voor u ligt is de Commodore 64-versie van het boek. Alle programma's in dit boek hebben we getest op een Commodore 64 en de programmatekst is rechtstreeks vanuit het geheugen van de Commodore op een letterwielprinter afgedrukt en zo in dit boek opgenomen. U kunt er dus van uitgaan dat alle programma's foutloos zijn. Tik ze dan ook precies zo in zoals ze in het boek staan beschreven.

Verder bevat de Nederlandse uitgave hier en daar nog wat nadere uitleg en suggesties hoe u de programma's kunt uitbreiden door te werken met kleuren en het opvullen van vlakken. De Commodore-programma's die de auteur in zijn boek heeft opgenomen gaan er vanuit dat de gebruiker beschikt over Simon's BASIC, omdat in deze programma's voor het inschakelen van de hoge-resolutiestand de opdracht HIRES 0,1 en voor het trekken van een lijn de opdracht LINE X1,Y1,X2,Y2,1 gebruikt worden. Dit is niet verwonderlijk, daar de standaard BASIC-versie die in de Commodore zit geen opdrachten kent voor het werken met hoge resolutie (320 bij 200 puntjes).

Om dit boek echter ook bruikbaar te maken voor diegenen die niet over Simon's BASIC beschikken hebben we in appendix 2 een BASIC-programma gegeven waarmee u acht machinetaalroutines kunt inlezen, zodat u vanuit uw standaard BASIC toch met hoge-resolutie-graphics kunt werken. Dit programma moet u helaas een keer intikken en op cassette of op diskette opslaan, maar u hebt dan ook 'gratis' de beschikking over net zulke snelle graphic-routines als in Simon's BASIC; u hoeft hiervoor dus geen extra BASIC-versie aan te schaffen.

Voor de assembleertaalenthousiasten hebben we deze acht graphic-routines in appendix 3 in een assembleertaal-listing afgedrukt. Het programma in appendix 2 is een lang programma, omdat het 2059 decimale machinetaalcodes bevat die u allemaal moet intikken. Maar ook dit programma, dat oorspronkelijk verschenen is in het blad 'Commodore Computing' van oktober 1983, hebben we zelf ingetoetst en getest. Ook hier kunt u ervan op aan dat het programma zoals het in appendix 2 staat goed is.

Alle programma's in dit boek zijn kort, dat wil zeggen: het langste is ongeveer 35 regels. Het resultaat ervan is echter boeiend en fascinerend om naar te kijken. Bovendien kunt u de programma's naar eigen fantasie verder verfraaien en uitbreiden; het aantal mogelijkheden is haast onbegrensd.

Het is eigenlijk een boek voor elke Commodore 64-bezitter. Diegenen die de veelal wiskundige uitleg bij de programma's teveel van het goede vinden, zullen alleen al door het intikken en draaien van de programma's verrukt zijn van het resultaat. Anderen, die zich nog iets van de sinus en cosinus uit hun schooltijd herinneren of die nu op school zitten, zullen weinig of geen moeite hebben met de theorie in dit boek. Leraren en leerlingen kunnen wellicht hun voordeel doen met de programma's voor het tekenen van functies. Ook het driedimensionaal tekenen wordt behandeld. Erg leuk zijn de vijf programma's waarmee in BASIC getekend kan worden zoals dat met LOGO ook kan. Dit geeft zeer fraaie grafieken, vooral de turtle-graphics. De laatste vijf programma's zijn toepassingeprogramma's, waarin 'graphics' de hoofdrol spelen.

Hopelijk hebt u net zoveel plezier met het draaien en verfraaien van deze programma's als wij hebben gehad bij de bewerking van dit boek.

Nok van Veen
december 1984

Voorwoord

Door het toenemende gebruik van microcomputers maken de grafische toepassingen een sterke ontwikkeling door. Het beeldscherm is tegenwoordig, veel meer dan de printer, het afdrukapparaat bij uitstek. Grafische toepassingen (graphics) vinden we bij computerspelletjes, bij computerkunst en in toenemende mate op de werkplek van ingenieurs, ontwerpers, architecten en anderen die zich bezighouden met Computer Ondersteund Ontwerpen (Computer Aided Design).

De geïnteresseerde leek is vaak verrukt van de mooie 'plaatjes', die bij demonstraties getoond worden, en denkt dat hiervoor hele ingewikkelde programma's nodig zijn. Iemand, die iets van wiskunde afweet, weet hoe een functie eruitziet en kan doorgaans zonder veel moeite grafische programma's maken.

Ik heb voor het Zwitserse Computertijdschrift 'MIKRO+KLEINCOMPUTER' een cursus 'programmeren met hoge resolutie' geschreven, die in een aantal afleveringen van het blad is verschenen. Al direct na het verschijnen van het eerste artikel werd ik werkelijk overstelpt met enthousiaste reacties van de lezers. Dit heeft ons aangezet tot het maken van dit boek.

Veel van de grafische toepassingsprogramma's worden geschreven voor een bepaald merk en type microcomputer of voor een bepaalde plotter. Wie een dergelijk programma wil herschrijven voor een ander merk microcomputer zal veel moeilijkheden ondervinden en vaak zelfs zijn pogingen staken.

De veertig programma's in dit boek zijn geschreven in standaard BASIC en gebruiken geen POKE- en PEEK-opdrachten. Bovendien worden slechts twee grafische opdrachten gebruikt, te weten het inschakelen van de hoge-resolutiestand en het verbinden van twee punten door een rechte lijn. De illustraties die als voorbeeld dienen van hetgeen de programma's tekenen zijn gemaakt op de miniplotter van de Sharp PC-1500.

Wij hopen dat de lezers veel plezier aan deze programma's zullen beleven en dat zij voor velen een aanmoediging zullen zijn om het terrein van de computergraphics verder te verkennen. Mijn dank gaat uit naar de heer H.J. Ottenbacher van Micro+Kleincomputer; zonder hem zou dit boek nooit het daglicht hebben gezien.

Marcel Sutter
voorjaar 1984

Inhoud

1 Graphics

In de High-Resolution-Graphic-stand bestaat het beeldscherm van de Commodore 64 uit 320 (40×8) puntjes horizontaal en 200 (25×8) puntjes verticaal. Vanuit een programma kunnen we elk van deze 64000 puntjes 'aan-en-uit' zetten en kleuren. In standaard Commodore 64-BASIC is dit echter nogal ingewikkeld, omdat hierin geen graphic-opdrachten (voor het tekenen van lijnen, crikels, e.d.) voorhanden zijn. Andere BASIC-versies voor de Commodore 64, zoals bijvoorbeeld Simon's BASIC, bevatten dergelijke grafische opdrachten wel. In dit boek worden eigenlijk maar twee van deze grafische opdrachten gebruikt en wel:

- Kies Hoge Resolutie stand
- Verbind punt P1 met P2

Deze twee opdrachten komen in de tekst voor als

- HIRES 0,1 èn
- LINE X1,Y1,X2,Y2,1

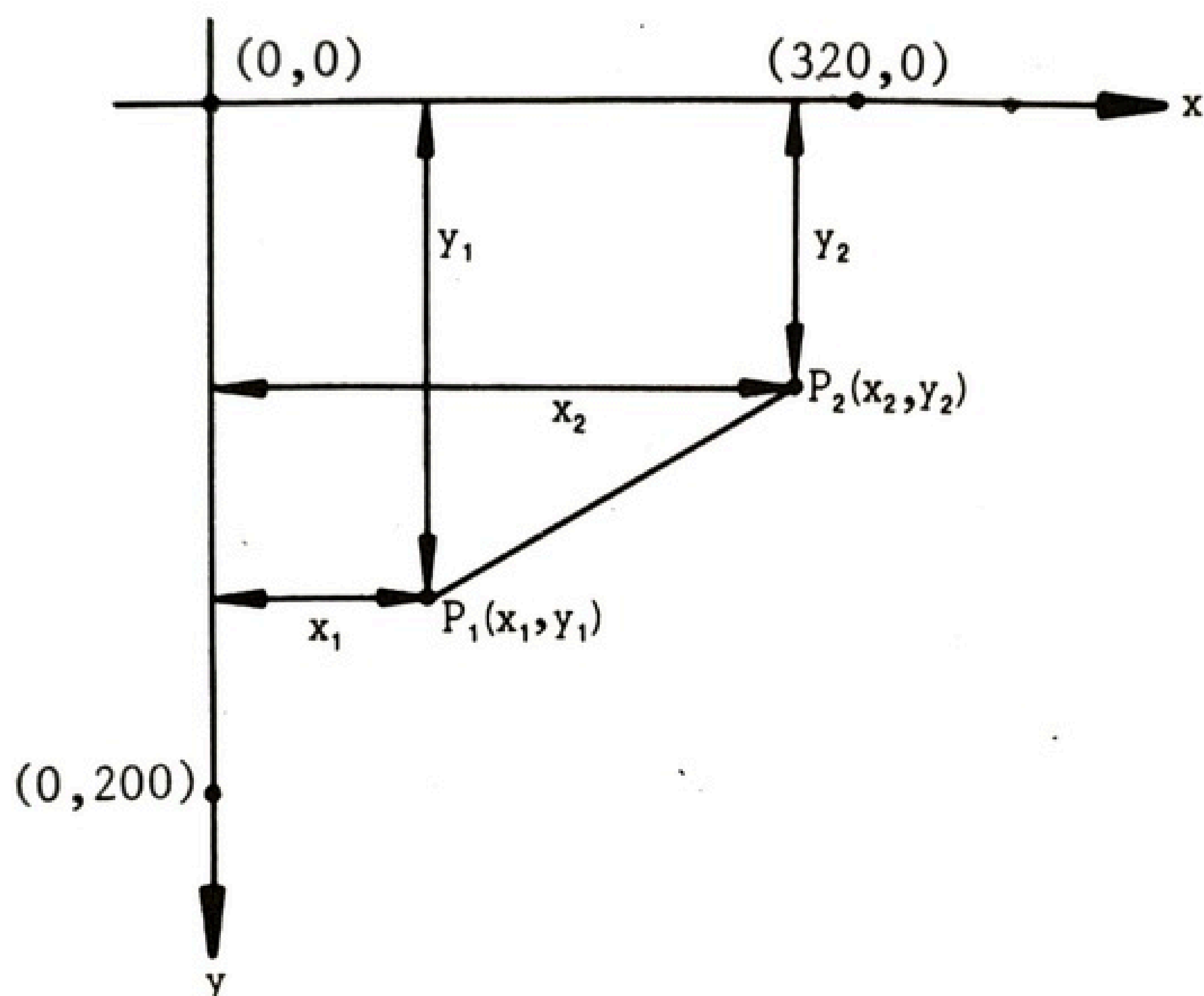
Dit zijn twee opdrachten uit Simon's BASIC. Simon's BASIC bevat nog veel meer grafische opdrachten, onder andere voor het tekenen van cirkels en het kleuren van vlakken. Van deze opdrachten maken we in dit boek echter geen gebruik om de programma's zo eenvoudig en zo duidelijk mogelijk te maken.

De opdracht HIRES 0,1 schakelt de Commodore 64 over op HRG (High Resolution Graphics). De nul geeft de kleur aan waarin getekend gaat worden (in dit geval zwart). De tweede parameter (1) geeft de achtergrondkleur aan (in dit geval wit). Bij HRG geeft 'zwart op wit' de scherpste tekeningen.

De opdracht LINE X1,Y1,X2,Y2,1 tekent een lijn tussen het beginpunt (X1,Y1) en het eindpunt (X2,Y2). De parameter 1 is het teken voor de computer om de lijn te trekken. Dezelfde opdracht met 0 als waarde voor de parameter zal de lijn weer uitwissen.

Voor diegenen die niet over Simon's BASIC of een andere uitgebreide Commodore-BASIC beschikken geven we in Appendix 1 aan hoe het effect van de HIRES- en LINE-opdrachten in standaard Commodore-BASIC-opdrachten gerealiseerd kan worden.

In de komende programma's gebruiken we dus horizontaal maximaal 320 puntjes en verticaal maximaal 200 puntjes. De oorsprong (nulpunt) van ons HRG-coördinatensysteem ligt in de linkerbovenhoek (HOME-positie) van het beeldscherm. De X-as loopt vanuit deze oorsprong naar rechts, de Y-as wijst naar beneden. Dit is verwarrend, daar wij gewend zijn de Y-as naar boven te laten wijzen, pas dus op!



Bij HRG onderscheiden we puntgraphics en lijn- (of vector-) graphics.

Puntgraphics

Bij puntgraphics worden de coördinaten x en y van een bepaald punt berekend en wordt dit punt door een bepaalde graphic-opdracht (bijvoorbeeld SET x,y) op het scherm zichtbaar. Willen we een dergelijk punt weer onzichtbaar maken, dan is daar een andere opdracht (bijvoorbeeld RESET x,y) voor nodig. In dit boek zullen we echter voor het tekenen van figuren niet van deze puntgraphics-techniek gebruik maken.

Appendix 2 bevat een BASIC-programma voor het inlezen van acht hoge-resolutie-routines in machinetaal. Twee van deze routines doen hetzelfde als HIRES en LINE. De gebruikers van standaard Commodore 64 BASIC zullen deze routines in plaats van HIRES en LINE moeten gebruiken. Kijk daarom alvast even in Appendix 2.

Lijngraphics

Bij lijngraphics worden door een bepaalde opdracht, die als machineroutine in de microcomputer aanwezig is, razendsnel twee berekende punten $P_1(X_1,Y_1)$ en $P_2(X_2,Y_2)$ door een rechte lijn met elkaar verbonden. Zo'n rechte verbindingslijn wordt opgebouwd uit een groot aantal kleine horizontale en verticale lijnstukjes. Op een afstand van zo'n 50 cm van het scherm lijken deze verbindingslijnen inderdaad recht.

In de in dit boek opgenomen programma's gebruiken we, zoals gezegd, slechts de volgende twee graphic-opdrachten:

'Kies Hoge Resolutie stand'	: HIRES 0,1	èn
'Verbind P1 met P2'	: LINE X1,Y1,X2,Y2,1	

Bewust hebben wij ons tot het gebruik van slechts deze twee opdrachten beperkt en geen gebruik gemaakt van de vaak zeer fraaie graphic-opdrachten die beschikbaar zijn. Ook hebben we de verleiding tot het gebruik van kleuren weerstaan. Hierdoor blijven de programma's klein en wordt de aandacht niet onnodig afgeleid. U kunt zelf de programma's naar believen met kleuropdrachten verfraaien.

Structuur van een HRG-programma

In alle programma's gebruiken we steeds dezelfde variabelen en dezelfde programmastructuur. De verschillende variabelen betekenen:

X1,Y1	coördinaten van het beginpunt
X2,Y2	coördinaten van het eindpunt
U,V	oorsprong van het wiskundige coördinatensysteem; dikwijls is dit het midden van het beeldscherm (160,100)
H	de waarde 0.5 voor het afronden op helen
K	de vermenigvuldigingsfactor voor functiewaarden
W,W1	hoeken bij trigoniometrische functies
RD	het getal $\pi/180$ voor het omrekenen van graden in radialen

Een grof structuurdiagram voor de programma's is op de volgende bladzijde weergegeven.

begin programma	
graphic-stand kiezen	
variabelen U,V,H,K,RD,enz. vastleggen	
punt P1(X1,Y1) vastleggen	
doe zolang nodig	
	nieuw punt P2(X2,Y2) berekenen
	verbindt P1 met P2
	punt P2 wordt punt P1 (X1=X2 Y1=Y2)
einde programma	

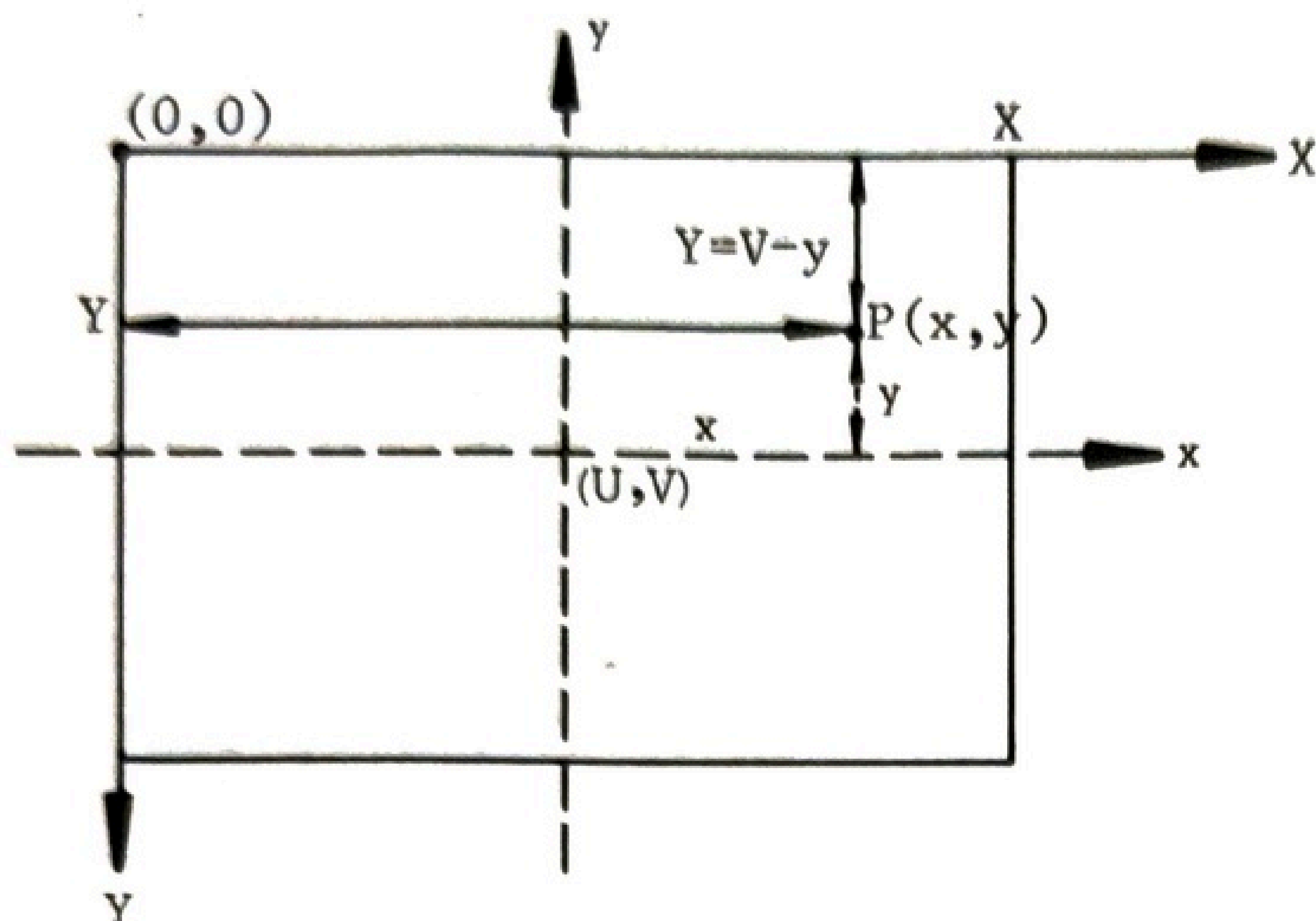
In alle programma's gebruiken we twee belangrijke transformatieformules.

Voor het HRG-coördinatensysteem van de Commodore is de linkerbovenhoek van het scherm de oorsprong. Wijzelf gebruiken echter bijna altijd een coördinatensysteem met de oorsprong in het midden van het scherm. Om de beeldschermcoördinaten X,Y (zoals de Commodore ze gebruikt) te berekenen uit de coördinaten x,y zoals wij ze gebruiken hanteren we de transformatieformules:

X = INT(U + x + H) : Y = INT(V - y + H)

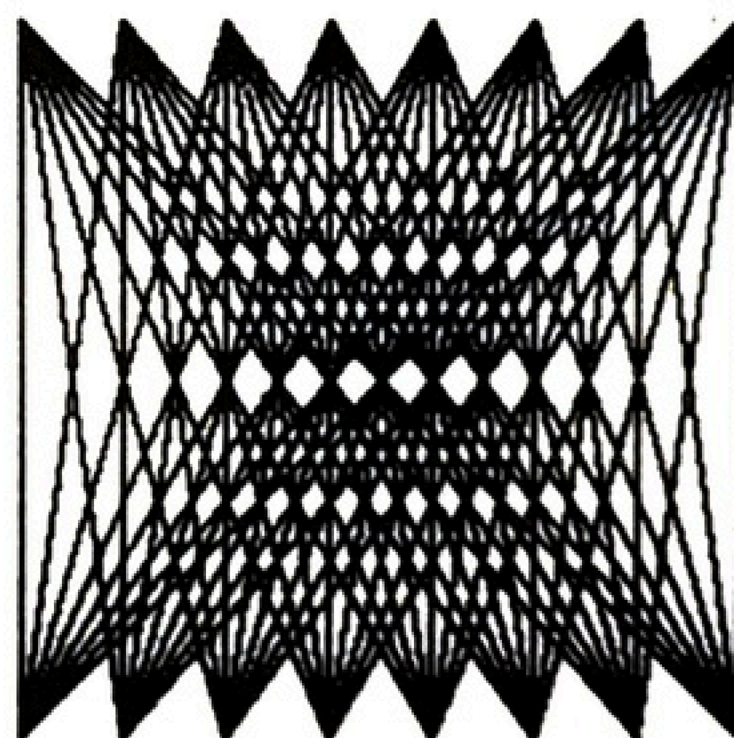
Bekijk de figuur op p.5 eens. Duidelijk is dat elk punt p(x,y) in ons coördinatenstelsel (gestippeld assenkruis) door bovenstaande formules wordt getransformeerd (overgebracht) naar een punt P(X,Y) in het coördinatenstelsel van de Commodore. Hierbij hoeft onze oorsprong (U,V) natuurlijk niet altijd precies in het midden van het beeld te liggen.

Genoeg theorie, nu de programma's. De ideeën voor de programma's hebben wij gekregen bij het doorbladeren van Amerikaanse, Duitse en Franse computertijdschriften. Alle programma's zijn echter eigen creaties of bewerkingen en vereenvoudigingen van reeds gepubliceerde programma's.



Programma 1 tekent een 'diagonaalweb'. Elk van de acht bovenste punten wordt door een rechte lijn verbonden met elk van de onderste acht punten.

```
100 REM PROGRAMMA 1 DIAGONAALWEB
110 PRINT CHR$(147)
120 HIRES 0,1
130 Y1=0 : Y2=200
200 FOR X1=0 TO 320 STEP 40
210 :  FOR X2=0 TO 320 STEP 40
220 :     LINE X1,Y1,X2,Y2,1
230 :  NEXT X2
240 NEXT X1
300 GET A$ : IF A$="" THEN 300
310 END
```



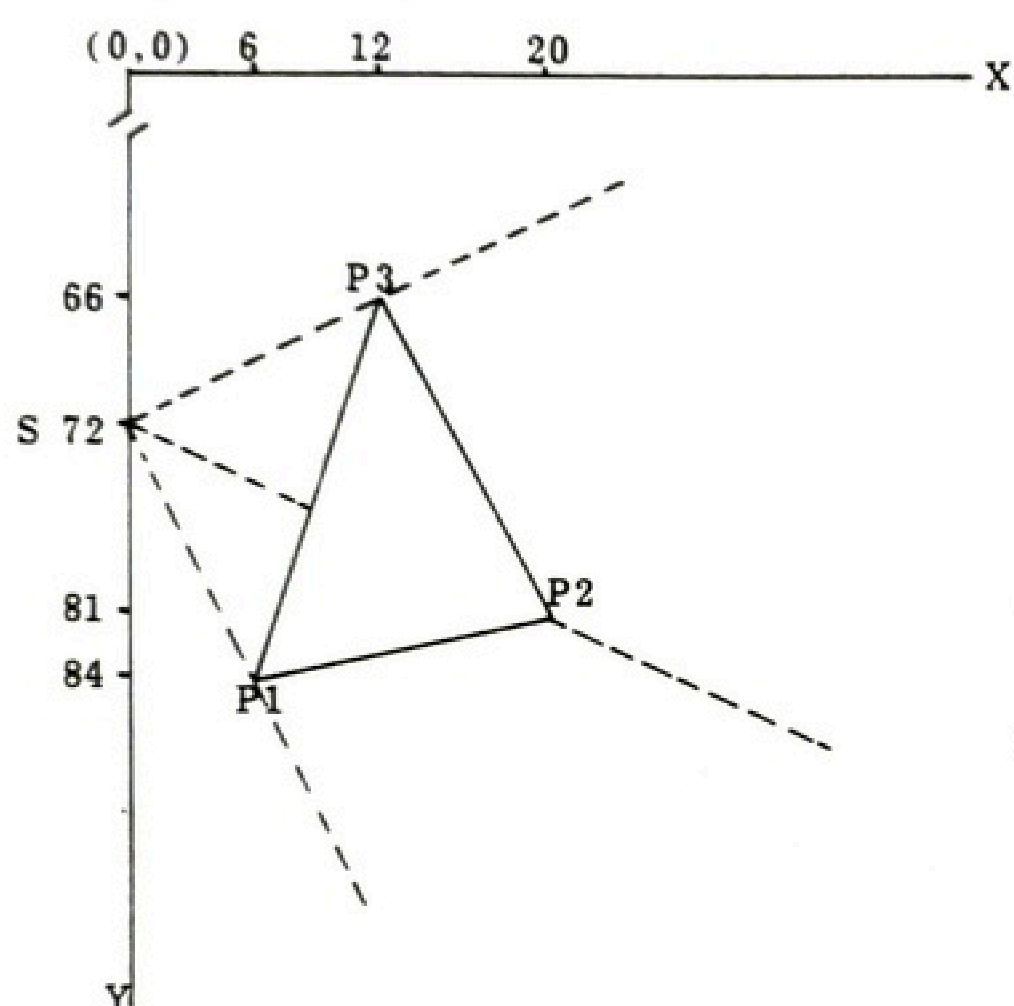
Kijk in Appendix 2 hoe u HIRES en LINE door machinetaalroutines kunt vervangen.

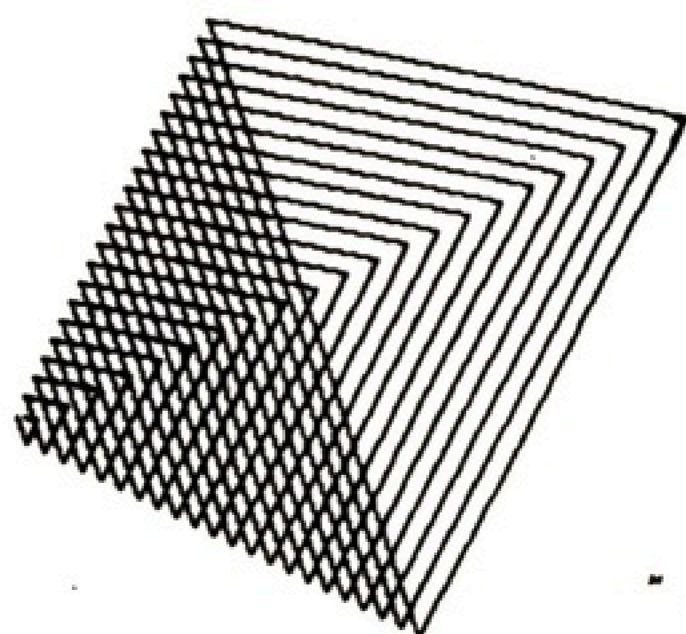
Programma 2 is een fraai voorbeeld van het Moiree-effect. Geniet ervan.



```
100 REM PROGRAMMA 2 MOIREE-EFFEKT
110 PRINT CHR$(147)
120 HIRES 0,1
130 FOR J=0 TO 200 STEP 8
140 :  LINE 60,0,J+60,200,1
150 :  LINE 60,0,260,(200-J),1
160 NEXT J
170 FOR J=0 TO 200 STEP 8
180 :  LINE 260,200,60,200-J,1
190 :  LINE 260,200,J+60,0,1
200 NEXT J
210 GET A$ : IF A$="" THEN 210
220 END
```

Programma 3 tekent een reeks driehoeken in perspectief. Het 'centrum van vermenigvuldiging' heeft de coördinaten S(0,72). De drie hoekpunten van de driehoek die vermenigvuldigd wordt zijn P1(6,84), P2(20,81) en P3(12,66). Alle coördinaten zijn beeldschermcoördinaten (oorsprong links bovenaan). De drie richtingen van vermenigvuldiging (richtingsvectoren) zijn $\overrightarrow{SP_1}$, $\overrightarrow{SP_2}$ en $\overrightarrow{SP_3}$. Deze worden in het programma als SX(1);SY(1), SX(2);SY(2) en SX(3);SY(3) vastgelegd. De vermenigvuldigingsfactor K is de lusvariabele van de buitenste FOR-NEXT lus. K loopt van 0 tot 11 in stappen van 0.5. Hieronder zien we de uitgangssituatie, het resultaat van het programma en het programma zelf.

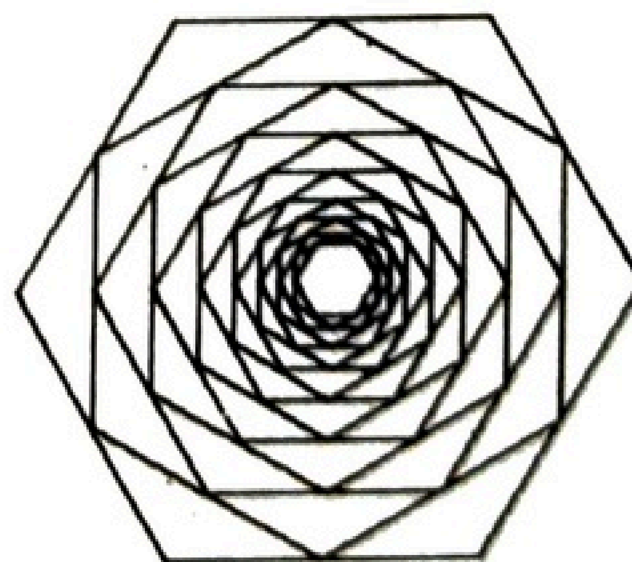




```
100 REM PROGRAMMA 3  DRIEHOEKEN IN
                     PERSPECTIEF
110 PRINT CHR$(147)
120 HIRES 0,1
130 DIMX(3),Y(3),SX(3),SY(3)
140 FOR J=1 TO 3
150 :  READ SX(J),SY(J)
160 NEXT J
170 X(0)=0 : Y(0)=72
180 FOR K=0 TO 11 STEP 0.5
190 :  FOR J=1 TO 3
200 :     X(J)=X(0)+K*SX(J)
210 :     Y(J)=Y(0)+K*SY(J)
220 :  NEXT J
230 :  LINE X(1),Y(1),X(2),Y(2),1
240 :  LINE X(2),Y(2),X(3),Y(3),1
250 :  LINE X(3),Y(3),X(1),Y(1),1
260 NEXT K
270 GET A$ : IF A$="" THEN 270
280 END
290 :   DATA 6,12,20,9,12,-6
```

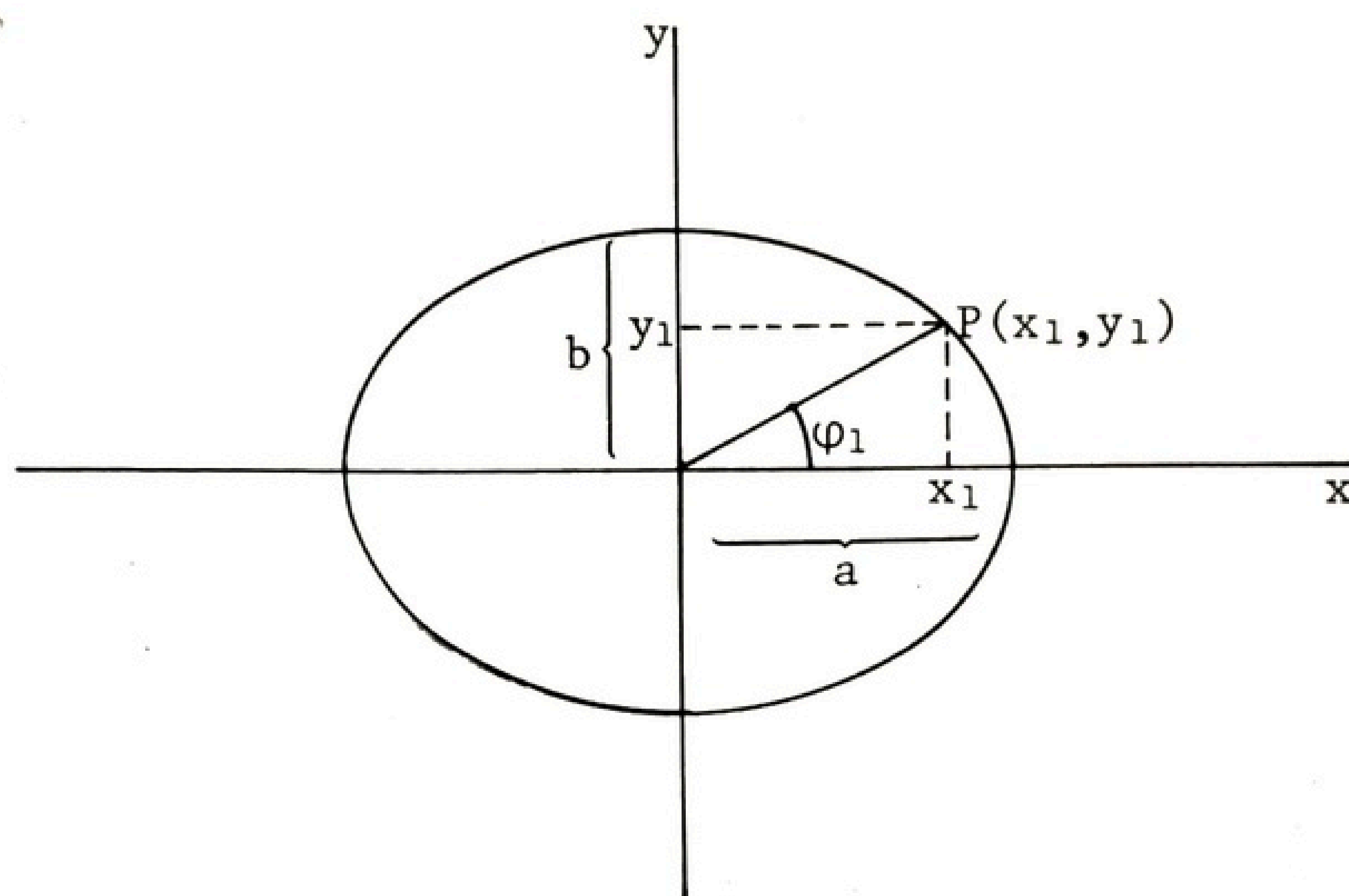
Programma 4 tekent een reeks ingeschreven regelmatige zeshoeken. Behalve bij de eerste zeshoek, liggen alle zes de hoeken van elke zeshoek precies in het midden van een zijde van de omgeschreven zeshoek. De coördinaten van de hoekpunten van de eerste (buitenste) zeshoek worden met trigoniometrische functies (sinus- en cosinusfunctie) berekend. De hoek die hierbij als argument van de functies gebruikt wordt doorloopt de waarden 0°, 60°, 120°, 180°, 240°, 300° en 360°. De middelpunten van de zijden van een zeshoek worden aangegeven met MX(0);MY(0) t/m MX(5);MY(5). Dit worden de hoekpunten van de volgende zeshoek.

```
100 REM PROGRAMMA 4 INGESCHREVEN
                    ZESHOEKEN
110 PRINT CHR$(147)
120 HIRES 0,1
130 DIM X(6),Y(6),MX(6),MY(6)
140 U=160:V=100:R=100:H=0.5:W=60*π/180
150 FOR J=0 TO 6
160 :  W1=J*W
170 :  X(J)=INT(U+R*COS(W1)+H)
180 :  Y(J)=INT(V-R*SIN(W1)+H)
190 NEXT J
200 FOR N=1 TO 20
210 :  FOR J=0 TO 5
220 :    LINE X(J),Y(J),X(J+1),Y(J+1),1
230 :  NEXT J
240 :  FOR K=0 TO 5
250 :    MX(K)=INT((X(K)+X(K+1))/2+H)
260 :    MY(K)=INT((Y(K)+Y(K+1))/2+H)
270 :  NEXT K
280 :  MX(6)=MX(0) : MY(6)=MY(0)
290 :  FOR J=0 TO 6
300 :     X(J)=MX(J) : Y(J)=MY(J)
310 :  NEXT J
320 NEXT N
330 GET A$ : IF A$="" THEN 330
340 END
```



Programma 5 tekent alle diagonalen in een n-hoek. De n hoekpunten liggen op de omtrek van een ellips of van een cirkel afhankelijk van de waarden die we in het begin van het programma voor de halve lange as a en halve kleine as b kiezen. Voor de berekening van de coördinaten van de punten op de omtrek van de ellips gebruiken we niet de (cartesische) vergelijking $(x^2/a^2) + (y^2/b^2) = 1$, maar de parametervorm $x = a.\cos\varphi$ en $y = b.\sin\varphi$.

Als de parameter φ loopt van 0° tot 360°, dan beschrijft het daaraan toegevoegde punt P(x,y), met $x = a\cos\varphi$ en $y = b\sin\varphi$, precies de omtrek van een ellips. Hieronder zien we een bepaald punt $P(x_1,y_1)$ op de omtrek van de ellips met de daarbij horende waarde van de parameter φ_1. Voor dat punt geldt: $x_1 = a\cos\varphi_1$ en $y_1 = b\sin\varphi_1$.



Als $x_1 = a\cos\varphi_1$ en $y_1 = b\sin\varphi_1$ dan geldt ook

$x_1^2 = a^2\cos^2\varphi_1$ en $y_1^2 = b^2\sin^2\varphi_1$ en ook

$\frac{x_1^2}{a^2} = \cos^2\varphi_1$ en $\frac{y_1^2}{b^2} = \sin^2\varphi$, dus dan is

$$\frac{x_1^2}{a^2} + \frac{y_1^2}{b^2} = \cos^2\varphi_1 + \sin^2\varphi_1$$

en omdat $\cos^2\varphi_1 + \sin^2\varphi_1$ gelijk is aan 1 geldt $\frac{x_1^2}{a^2} + \frac{y_1^2}{b^2} = 1$ en hebben we de cartesische relatie tussen x_1 en y_1 afgeleid uit de parametervorm.

Voor het berekenen van de coördinaten van het j-de hoekpunt (X(J),Y(J)) verdelen we de maximale middelpuntshoek van 360° in n gelijke hoeken van elk 360/n graden.

Als A de lengte van de halve lange as is èn
B de lengte van de halve korte as èn
N het aantal hoekpunten van de n-hoek èn
W gelijk is aan 360/N èn
W1 de hoek die hoort bij het j-de hoekpunt èn
X(J) de x-coördinaat van het j-de hoekpunt t.o.v. het middelpunt van de ellips èn
Y(J) de y-coördinaat van het j-de hoekpunt t.o.v. het middelpunt van de ellips

dan zou je in een BASIC-programma de coördinaten X(J) en Y(J) als volgt kunnen berekenen:

```
W     = 360/N : W1 = J*N
X(J)  = INT(A*COS(W1))
Y(J)  = INT(B*SIN(W1))
```

Als we dit doen maken we twee fouten. De eerste fout is dat de hoek W1, als argument van de COSinus- en de SINusfunctie, in graden is uitgedrukt, terwijl BASIC vereist dat het argument van deze functies in radialen wordt uitgedrukt. De tweede fout is dat het HRG-systeem van de Commodore de oorsprong van het coördinatenstelsel voor het scherm in de linkerbovenhoek van het scherm verwacht en niet in het middelpunt van de ellips. Wat we dus nog moeten doen is:

a. bereken W1 in radialen èn
b. transformeer X(J) en Y(J) naar beeldschermcoördinaten.

We weten dat een hoek van 360° overeenkomt met 2π radialen, waarin π = 3,14159..... . Dit betekent dat een hoek van 1 graad overeenkomt met een hoek van 2π/360 of π/180 radialen. De transformatieformules voor de transformatie van 'onze' coördinaten naar beeldschermcoördinaten hebben we op p.4 gegeven. De drie BASIC-opdrachten voor het berekenen van de beeldschermcoördinaten van het j-de hoekpunt van de n-hoek worden nu:

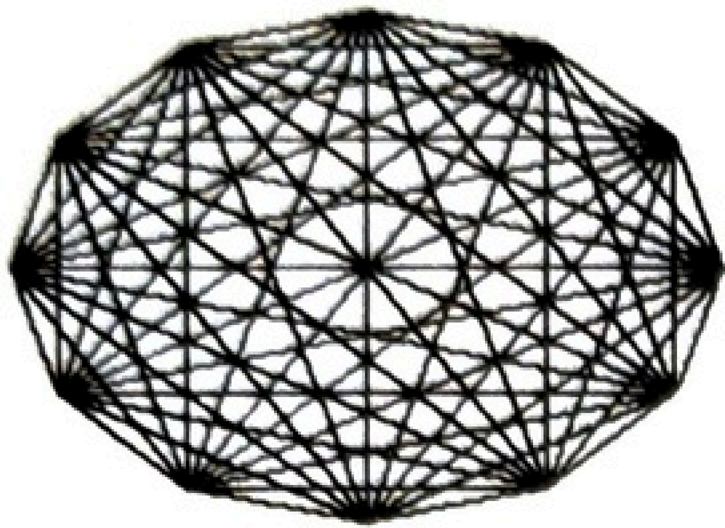
```
W = (360/N)*π/180 : W1 = J*W
X(J) = INT(U + A*COS(W1) + H)
Y(J) = INT(V - B*SIN(W1) + H)
```

We gebruiken de π-toets van het Commodore-toetsenbord!

Een andere manier om π in BASIC te berekenen is met de inverse tangensfunctie: π = 4*ATN(1).

In het navolgende programma komen we bovenstaande opdrachten

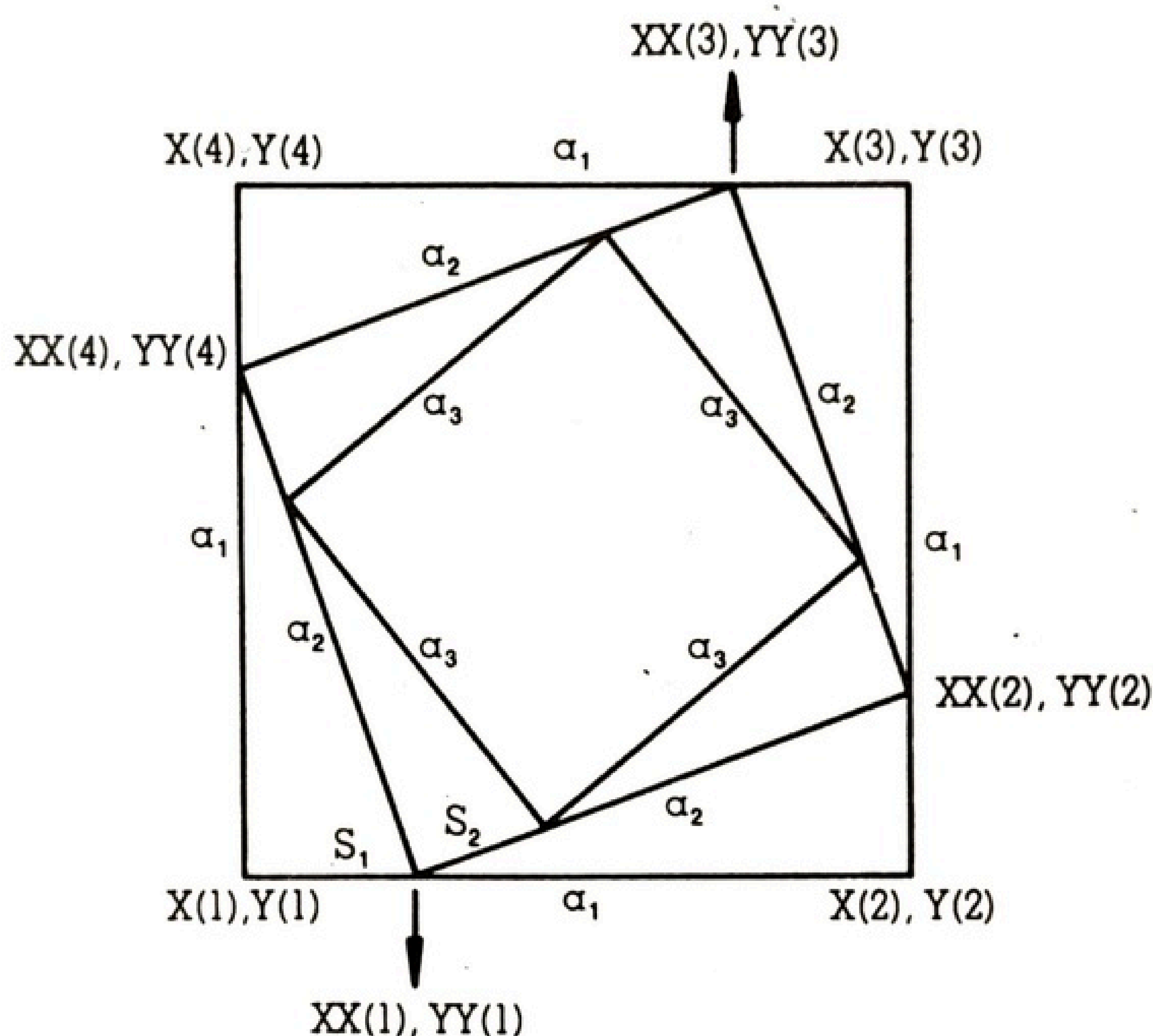
tegen. Kiezen we voor A en B dezelfde waarde dan krijgen we een regelmatige n-hoek met al zijn diagonalen. Bovendien is de ellips dan een cirkel geworden.



```
100 REM PROGRAMMA 5 DIAGONALEN IN EEN
                      N-HOEK
110 PRINT CHR$(147)
120 INPUT "HALVE GROTE AS A<=160"; A
130 INPUT "HALVE KLEINE AS B<=100"; B
140 INPUT "HOEVEEL HOEKPUNTEN N<25"; N
150 PRINT CHR$(147)
160 HIRES 0,1
170 DIM X(N),Y(N)
180 U=160:V=100:H=0.5:W=(360/N)*π/180
190 FOR J=0 TO N-1
200 :  W1=J*W
210 :  X(J)=INT(U+A*COS(W1)+H)
220 :  Y(J)=INT(V-B*SIN(W1)+H)
230 NEXT J
240 FOR I=0 TO N-2
250 :  FOR J=I+1 TO N-1
260 :    LINE X(I),Y(I),X(J),Y(J),1
270 :  NEXT J
280 NEXT I
290 GET A$ : IF A$="" THEN 290
300 END
```

Programma 6 is wat veeleisender in die zin dat het wat voorbereiding nodig heeft.

We willen een reeks ingeschreven vierkanten zo tekenen dat de hoekpunten van een ingeschreven vierkant uit de reeks op de zijden liggen van zijn voorganger. We zien deze situatie voor het eerste, tweede en derde vierkant van de reeks in de onderstaande tekening. De afstand tussen een hoekpunt van een vierkant en een hoekpunt van het volgende vierkant (S_1 en S_2) is steeds een vast deel van de zijde waarop de hoekpunten liggen. Zo is $S_1 = a_1/k$ en $S_2 = a_2/k$. In het algemeen is $S_n = a_n/k$, waarbij a_n de zijde is van het n-de vierkant in de reeks en k de verkleiningsfactor. De waarde voor k kunnen we zelf kiezen. k=16 geeft een fraaie tekening.



(X(1),Y(1)); (X(2),Y(2)); (X(3),Y(3)) en (X(4),Y(4)) zijn de coördinaten van de hoekpunten van een bepaald vierkant uit de reeks. Hoe berekenen we nu de coördinaten van de hoekpunten van het volgende ingeschreven vierkant ((XX(1),YY(1)); (XX(2),...)?

Voor het eerste hoekpunt van het eerste ingeschreven vierkant geldt

$$XX(1) = X(1) + \frac{X(2)-X(1)}{K}$$ èn

$$YY(1) = Y(1) + \frac{Y(2)-Y(1)}{K}$$

Ga na dat dit klopt in bovenstaande tekening.

Voor het tweede hoekpunt van het ingeschreven vierkant geldt:

$$XX(2) = X(2) + \frac{X(3)-X(2)}{K}$$ èn

$$YY(2) = Y(2) + \frac{Y(3)-Y(2)}{K}$$

In het algemeen geldt:

$$XX(J) = X(J) + \frac{X(J+1)-X(J)}{K}$$ èn

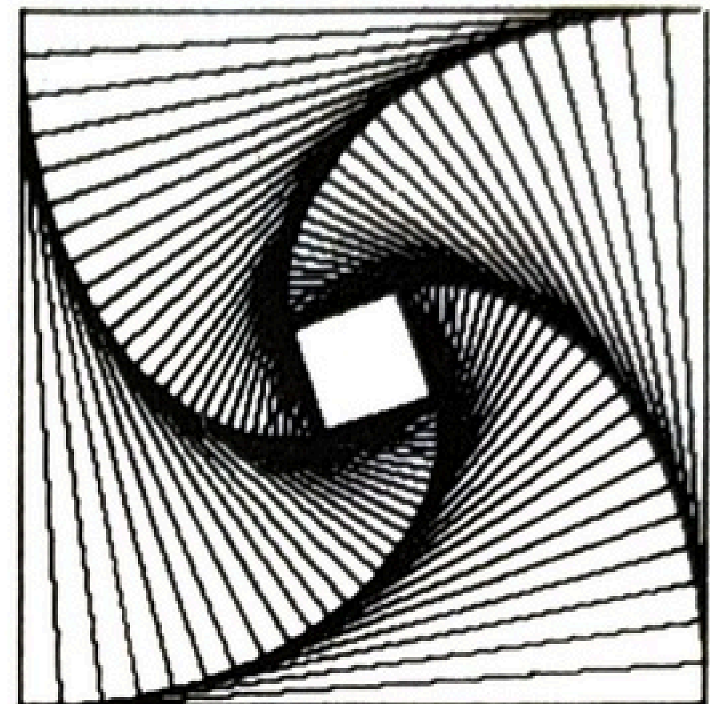
$$YY(J) = Y(J) + \frac{Y(J+1)-Y(J)}{K}$$

voor J = 1,2,3,4, waarbij X(5) = X(1) en
Y(5) = Y(1).

Als we het tweede vierkant getekend hebben, veranderen we XX(1) in X(1), YY(1) in Y(1), XX(2) in X(2), YY(2) in Y(2), enzovoorts, en tenslotte X(5) in X(1) en Y(5) in Y(1) en we gaan met dezelfde formules opnieuw XX(1),YY(1),... enz. berekenen, maar dan voor de hoekpunten van het volgende vierkant. We zien dit gebeuren in de regels 270 t/m 300 van het volgende programma. De regels 200 t/m 220 tekenen het vierkant, terwijl de regels 230 t/m 260 de hoekpunten van het volgende vierkant berekenen. Een structuurdiagram voor het programma ziet er zo uit:

begin programma ingeschreven vierkanten	
HRG-voorbereidingen	
lees waarde voor K in	
bepaal coördinaten voor het eerste vierkant	
voor het eerste t/m 40-ste vierkant doe	
	teken dit vierkant
	bepaal coördinaten voor het volgende vierkant
einde programma	

Hier komt het programma:



```
100 REM PROGRAMMA 6 INGESCHREVEN
                    VIERKANTEN
110 PRINT CHR$(147)
120 INPUT "TOETS K IN 1<K<20"; K
130 HIRES 0,1 : H=0.5
140 DIM X(5),Y(5),XX(5),YY(5)
150 FOR J=1 TO 5
160 :  READ X(J),Y(J)
170 NEXT J
180 PRINT CHR$(147)
190 FOR N=1 TO 40
200 : FOR J=1 TO 4
210 :   LINE X(J),Y(J),X(J+1),Y(J+1),1
220 : NEXT J
230 : FOR J=1 TO 4
240 : XX(J)=X(J)+INT((X(J+1)-X(J))/K+H)
250 : YY(J)=Y(J)+INT((Y(J+1)-Y(J))/K+H)
260 : NEXT J
270 : FOR J=1 TO 4
280 :    X(J)=XX(J) : Y(J)=YY(J)
290 : NEXT J
300 : X(5)=X(1) : Y(5)=Y(1)
310 NEXT N
320 GET A$ : IF A$="" THEN 320
330 END
340 :   DATA 60,200,260,200,260,0,60,0
350 :   DATA 60,200
```

U kunt dit programma als uitgangspunt voor een uitgebreider programma gebruiken. Verdeel het beeldscherm in, bijvoorbeeld, zes gelijke vierkanten (elk vierkant 50×50 puntjes). In elk van deze zes vierkanten tekent u, met bovenstaand programma, een reeks van ingeschreven vierkanten. Teken zes identieke reeksen, of kies steeds een andere waarde voor K of probeer ze ten opzichte van elkaar te laten draaien.

2 Grafieken van functies in cartesische vorm

In het eerste hoofdstuk hebben we alleen rechtlijnige patronen getekend, bijvoorbeeld het programma voor het tekenen van alle diagonalen in een regelmatige n-hoek.

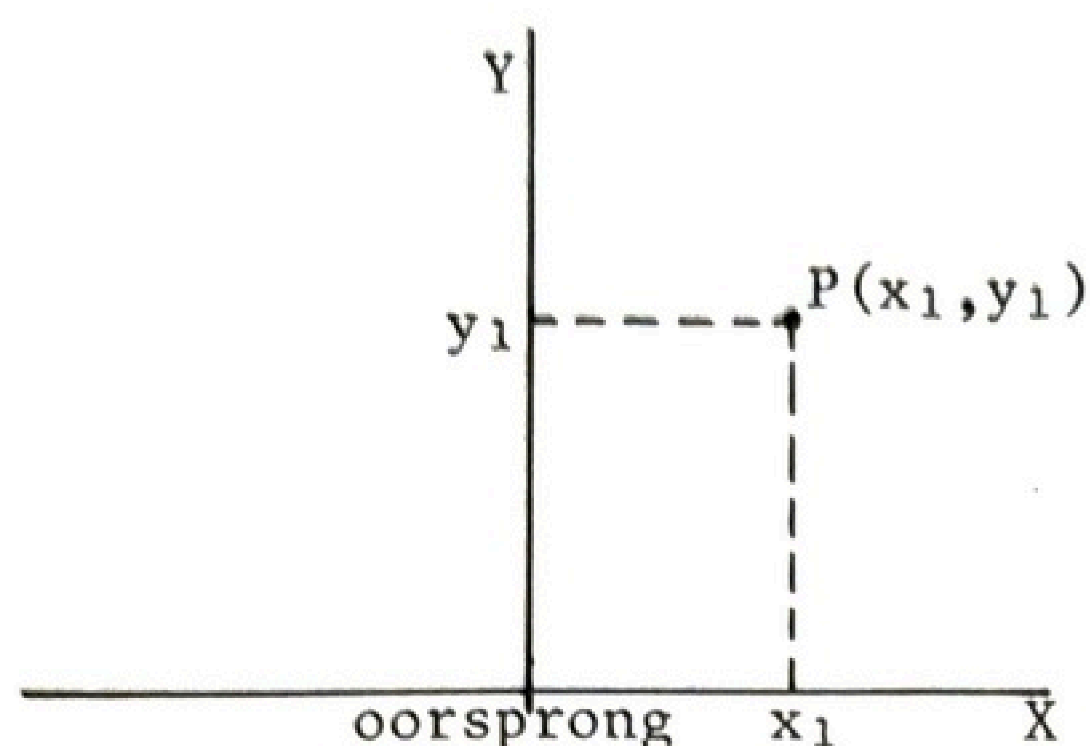
In dit en het volgende hoofdstuk zullen we ons uitvoerig bezig houden met het tekenen van de grafiek van een aantal continue en niet-continue functies. In de wiskunde noemen we de grafiek van een niet-lineaire functie een kromme. De grafiek van een lineaire functie (bijvoorbeeld de functie $y = 4x + 3$) noemen we een rechte. We kunnen de vergelijking van zo'n kromme op drie manieren formuleren:

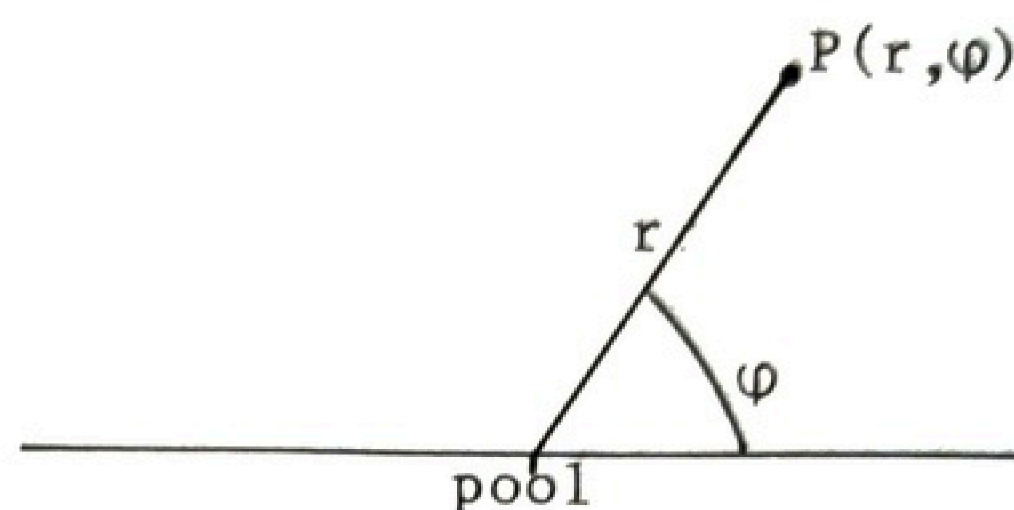
A	; met cartesische coördinaten	: $y = f(x)$
B	; met poolcoördinaten	: $r = f(\varphi)$
C	; of in parametervorm	: $x = f(t)$, $y = g(t)$

Niet-wiskundigen kennen vaak alleen de gewone (cartesische) vorm $y = f(x)$. Een voorbeeld van een niet-lineaire functie in cartesische vorm is de functie $y = 2x^2 - 3x + 4$. De grafiek van deze functie is een parabool.

De functie $r = 110.\cos(4\varphi)$ stelt een kromme in poolcoördinaten voor. Als we de hoek φ laten lopen van 1 tot 360 graden en we tekenen bij elke hoek φ onder die hoek een punt op afstand r van de oorsprong, dan krijgen we de bloemfiguur van pagina 40.
Deze vorm $r = 110.\cos(4\varphi)$ heet de poolcoördinatenvorm. Een punt in het platte vlak wordt nu niet gekenmerkt door de afstand van dat punt tot de x- en y-as (cartesisch), maar door de afstand tot de oorsprong (r) en de hoek φ die de x-as maakt met de lijn die dat punt met de oorsprong verbindt. Poolcoördinaten komen in hoofdstuk 3 aan de orde (zie tekening op p.16).

Er is nog een manier om de vergelijking van een kromme weer te geven en dat is de parametervorm. Hierbij worden de x- en y-coördinaat van een punt op de kromme afhankelijk gemaakt van een





Hetzelfde punt P in een
cartesisch coördinatenstelsel en een poolcoördinatenstelsel

bepaalde parameter. Een voorbeeld zijn de vergelijkingen

$$x = a.\cos(t) \quad \text{en} \quad y = a.\sin(t).$$

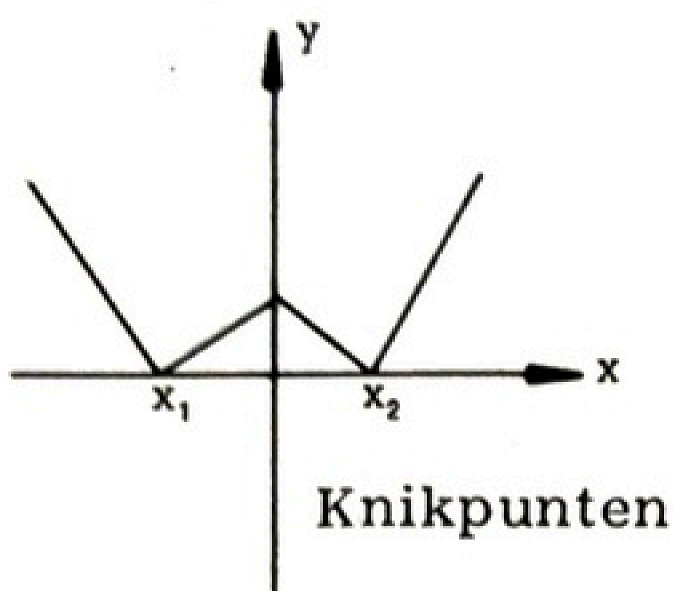
Als we t laten lopen van 0 tot 360°, beschrijven de daarbij horende punten (x,y) precies de omtrek van een cirkel met straal a. De cartesische vorm van deze cirkelvergelijking is $x^2 + y^2 = a^2$, die velen direct zullen herkennen als de 'cirkelvergelijking'.

In dit hoofdstuk houden we ons dus alleen bezig met het tekenen van grafieken van functies en relaties die door middel van cartesische coördinaten beschreven worden. De andere twee vormen komen in hoofdstuk 3 aan de orde.

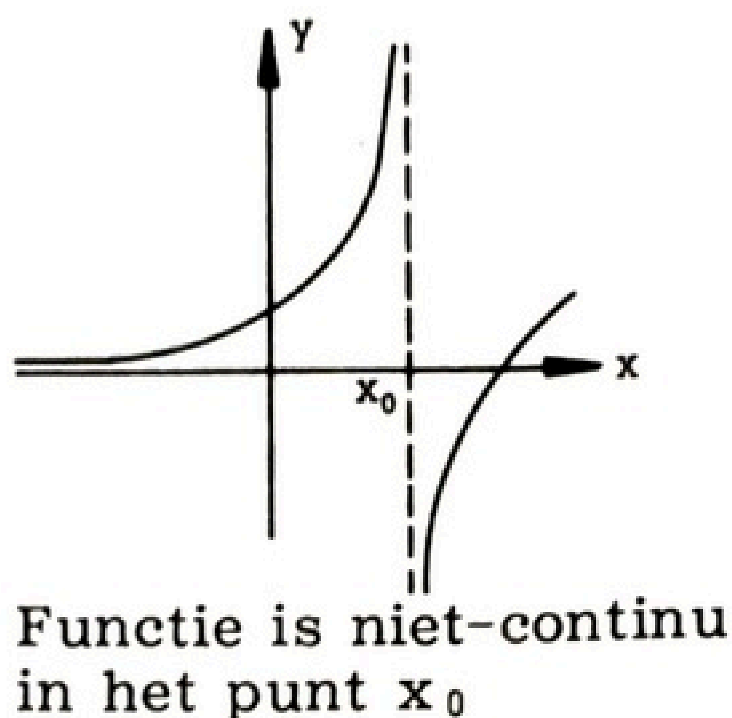
Continue functies

Een continue functie is een functie waarvan de grafiek in één vloeiende beweging van ons 'potlood', dat wil zeggen zonder het potlood van het papier te hoeven halen, getekend kan worden. Er mogen 'knikken' in voorkomen maar geen onderbrekingen (zie tekening op p.17).

De linkergrafiek kunnen we in één beweging, zonder het potlood van het papier te halen, tekenen; bij de rechter grafiek kan dat niet. De linker grafiek is de grafiek van een continue functie; de rechter van een niet-continue functie. Als we alleen links of alleen rechts van het punt x_0 kijken dan is de functie op die intervallen natuurlijk wel continu! De functie is alleen niet-continu in het punt x_0.



Knikpunten



Functie is niet-continu in het punt x_0

We willen nu een algemeen programma ontwikkelen dat, gegeven de grenzen a en b van een bepaald interval ($a \leq x \leq b$) de grafiek tekent van een willekeurige continue functie y = f(x). Mochten de x-as en de y-as in het gebied liggen waarin de grafiek van de functie wordt getekend, dan willen we dat ook beide assen door het programma getekend worden. Om het programma kort, maar toch algemeen, te houden brengen we de volgende vereenvoudigingen aan.

1. De vergelijkingen van de functie, die getekend moet worden,. wordt in een subroutine, vanaf regel 1000, opgenomen. De functiewaarde y = f(x), bij een bepaalde x-waarde, wordt berekend door op dat moment met de gewenste x-waarde de subroutine aan te roepen (GOSUB 1000). Er wordt dus niet gebruik gemaakt van de methode waarin de functie als tekst (INPUT-opdracht met stringvariabele) wordt ingelezen waarna het programma deze tekst zelf omzet in een goede BASIC functiedefinitie. Een andere mogelijkheid zou zijn om de functie met een DEF FN-opdracht in het programma te definiëren.

2. Op de coördinaatassen wordt niet automatisch een schaalverdeling aangebracht en ook wordt er geen tekst bij afgedrukt. Veel micro's met HRG-mogelijkheden kunnen niet tegelijkertijd tekenen en 'schrijven'. Wilt u toch graag weten wat bij een bepaalde waarde van x de functiewaarde (y) is, dan kunt u bijvoorbeeld na het tekenen van de grafiek de computer (op de printer) een lijstje met x- en y-waarden laten afdrukken.

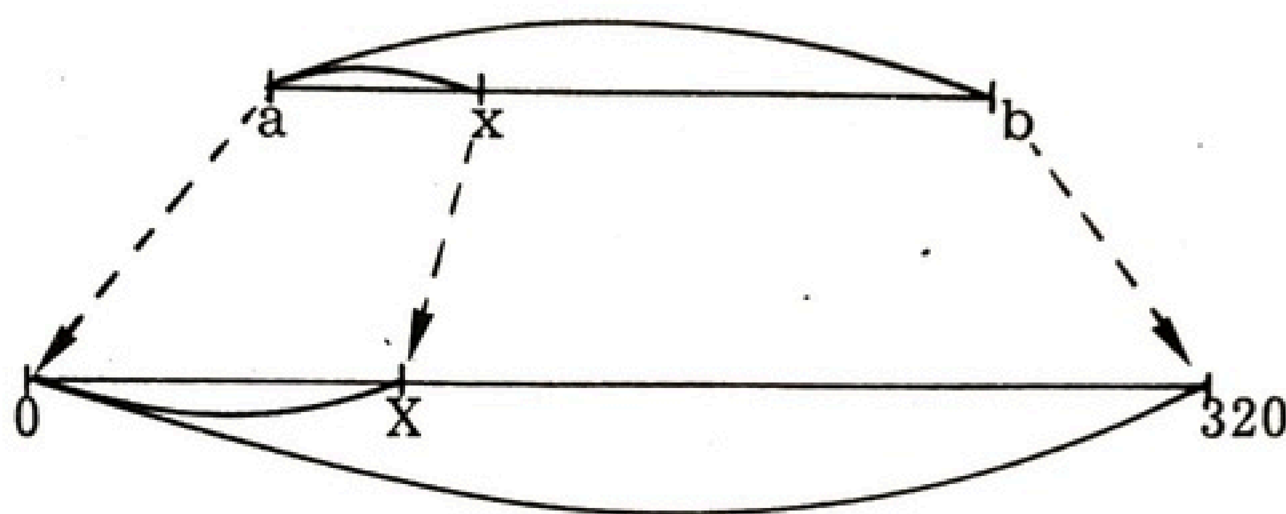
Terug naar het programma-ontwerp. In het eerste deel van het programma berekent de computer, na het inlezen van de waarden van de intervalgrenzen a en b, de grootste en de kleinste functiewaarde (y-waarde) in het opgegeven interval [a,b]. De waarde van HP

(Hoogste Punt) en LP (Laagste Punt) worden op het beeldscherm afgedrukt. We weten zo in welk gebied de computer gaat tekenen. Hierna vraagt het programma of we HP nog groter en of we LP nog kleiner willen maken om daarmee een fraaiere tekening te krijgen. Zo niet, dan toetsen we voor HP en LP dezelfde waarden in als het programma heeft berekend; anders toetsen we de door ons gewenste maximale en minimale functiewaarden in.

Het volgende programmadeel (regels 280-360) bevat de lus (FOR A TO B STEP DX) voor het tekenen van de grafiek. De coördinaten x,y van een punt op de grafiek moeten met de juiste transformatieformules omgezet worden in beeldschermcoördinaten X,Y.

Om het interval [a,b] voor te kiezen x-waarden ($a \leq x \leq b$) om te zetten in het HRG-interval [0,320] ($0 \leq X \leq 320$) gebruiken we de volgende evenredigheid:

$$(x-a) : (b-a) = X : 320 \quad \text{ofwel} \quad \frac{x-a}{x-b} = \frac{X}{320}$$



in BASIC:

```
X2 = INT(KX*(X-A)+0.5),   met KX = 320/(B-A).
```

Omdat BASIC in hoofdletters 'werkt' hebben we in deze formule voor x de variabele X genomen, voor X de variabele X2, voor a de variabele A en voor b de variabele B.

We doen dit ook voor het afbeelden van het functiebereik $LP \leq y \leq HP$ op het HRG-beeldbereik $0 \leq Y \leq 200$:

$$(HP-y) : (HP-LP) : 200 \quad \text{ofwel} \quad \frac{HP-y}{HP-LP} = \frac{Y}{200}$$

Dit geeft in BASIC:

```
Y2 = INT(KY*(HP-Y)+0.5)   met KY = 200/(HP-LP)
```

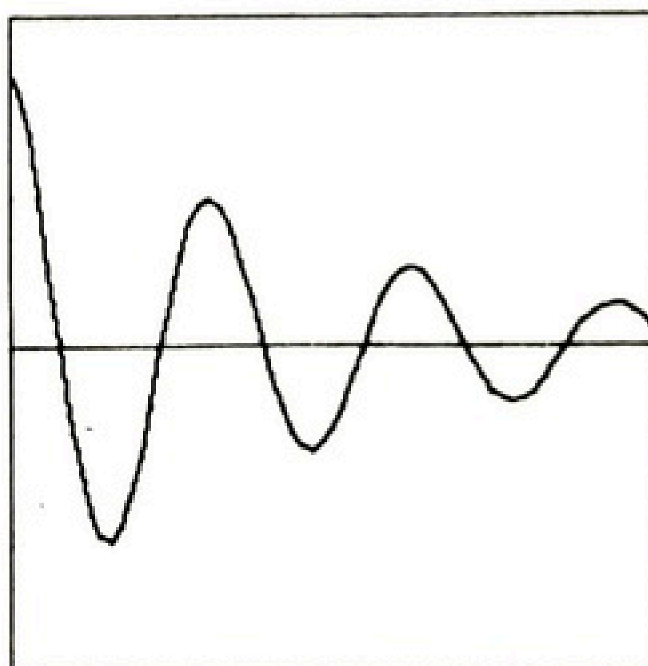
We zien deze berekeningen van X2 en Y2 in de regels 300 en 310. In de volgende regel worden twee, op deze wijze berekende, punten (X1,Y1) en (X2,Y2) door een recht lijntje met elkaar verbonden.

Vervolgens (regels 370-400) wordt de ligging van de x- en y-as bepaald. Kiezen we in bovenstaande transformatievergelijkingen voor X en Y de waarde nul, dan vinden we respectievelijk de ligging van de y-as en van de x-as (regels 370 en 390).

Met deze uitleg is de werking van het programma hopelijk te volgen.

Test u het programma 7 met willekeurige, maar continue, functies. In dit voorbeeldprogramma hebben we de functie $y = e^{-0.1x}.\cos(x)$ gekozen, hetgeen de grafiek van een gedempte trilling oplevert. Hieronder zijn enkele ideeën voor minder moeilijke functies.

functie	regel 1000	waarde voor a	waarde voor b
$y = \sin(x)$	Y = SIN(X)	0	2π (= 6,2832)
$y = x^2$	Y = X↑2	-4	+4
$y = e^x$	Y = EXP(X)	-3	+3
$y = x^3-2x^2-x$	Y = X↑3+2*X↑2-X		



```
100 REM PROGRAMMA 7 GRAFIEK VAN EEN
                    CONTINUE FUNCTIE
110 PRINT CHR$(147)
120 INPUT"LINKER-INTERVAL GRENS"; A
130 INPUT"RECHTER-INTERVAL GRENS"; B
140 IF A>B THEN C=A:A=B:B=C
150 HP=-100000:LP=100000:DX=(B-A)/64
160 FOR X=A TO B STEP DX
170 :   GOSUB 1000
180 :   IF Y>HP THEN HP=Y
190 :   IF Y<LP THEN LP=Y
200 NEXT X
210 PRINT "GROOTSTE Y-WAARDE:"; HP
220 PRINT "KLEINSTE Y-WAARDE:"; LP
230 INPUT"BOVENGRENS VOOR Y"; HP
240 INPUT"ONDERGRENS VOOR Y"; LP
250 PRINT CHR$(147)
260 HIRES 0,1
270 KX=320/(B-A):KY=200/(HP-LP):H=0.5
280 FOR X=A TO B STEP DX
290 :   GOSUB 1000
300 :   XX=INT(KX*(X-A)+H)
310 :   YY=INT(KY*(HP-Y)+H)
320 :   IF X=A THEN X1=XX:Y1=YY:GOTO360
330 :   X2=XX:Y2=YY
340 :   LINE X1,Y1,X2,Y2,1
350 :   X1=X2:Y1=Y2
360 NEXT X
370 X1=0:Y1=INT(KY*HP+H):X2=320:Y2=Y1
380 IF Y1>=0 AND Y1<=320
            THEN LINE X1,Y1,X2,Y2,1
390 X1=INT(KX*(-A)+H):Y1=0:X2=X1:Y2=200
400 IF X1>=0 AND X1<=320
            THEN LINE X1,Y1,X2,Y2,1
410 GET A$ : IF A$="" THEN 410
420 END
430 :
1000 Y=EXP(-0.1*X)*COS(X)
1010 RETURN
```

Nu volgen drie korte programma's.

Programma 8 is de eerste van deze drie. Dit programma tekent tien in fase verschoven sinuskrommen, allemaal in hetzelfde coördinatenstelsel. De vergelijking van het stelsel is

$$y = \sin(x+np), \quad \text{met als fase } p = \frac{\pi}{9} \ (20°).$$

We vinden de tien vergelijkingen door voor n de waarden 0,1,2 t/m 9 te kiezen. De lusvariabele J (regel 150) komt overeen met de HRG-beeldschermcoördinaat X2. Omdat we in dit programma de beeldschermcoördinaat X2, dat wil zeggen J, in stappen van 5 naar 320 laten lopen (regel 150), moeten we eerst de bij X2 behorende waarde voor x berekenen. Dan pas kunnen we de functiewaarde y = f(x) berekenen. We kunnen de evenredigheid

$$(x-a) : (b-a) = X2 : 320$$

natuurlijk ook gebruiken om x uit X2 te berekenen; hieruit volgt:

$$x = \frac{(b-a)}{320}.X2 + a.$$

In het onderstaande programma zijn de gekozen intervalgrenzen a = 0 en b = 2π, zodat bovenstaande vergelijking wordt:

$$x = \frac{2\pi}{320}.X2.$$

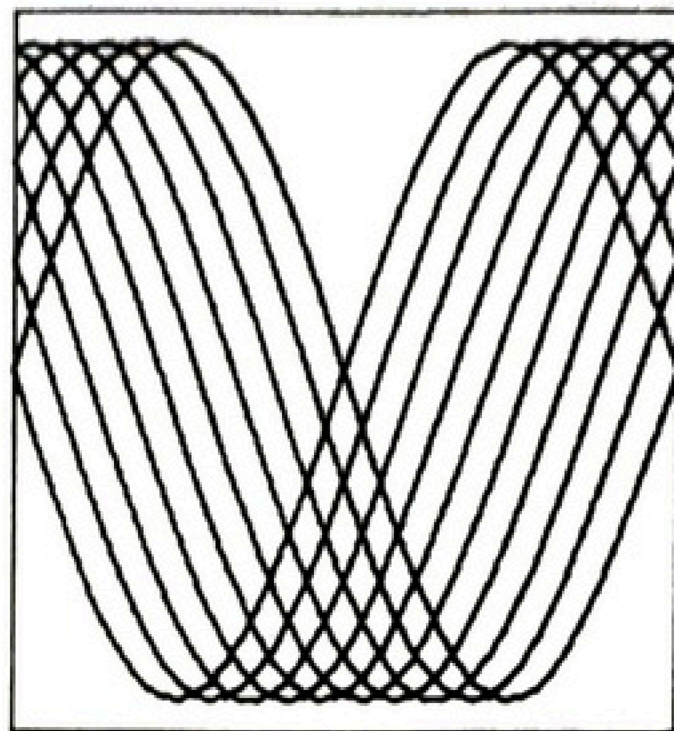
Als we $C = \frac{2\pi}{320}$ nemen, wordt in het programma het argument van de sinusfunctie dus C*X2+N*P, ofwel C*J+N*P.

In het programma wordt de waarde C*J als X berekend, dus zien we in regel 160 X+N*P als argument van de SINusfunctie. Natuurlijk moeten we de functiewaarde y = SIN(X+N*P) nog omzetten naar de beeldschermcoördinaat Y zodat we krijgen

Y2 = INT(V-K*SIN(X+N*P)+H).

Door voor de schaalconstante K de waarde 100 te kiezen (regel 130) krijgen we een mooie 'grote' tekening.

Wie met kleuren wil werken kan dit programma uitbreiden door de diverse krommen andere kleuren te geven. Iets moeilijker zal het zijn de 'banen' tussen de sinuskrommen in te kleuren. Probeer het eens.

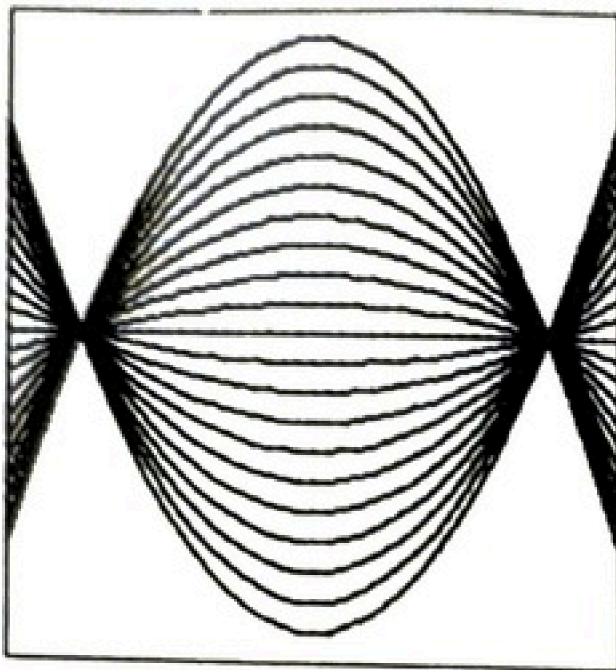


```
100 REM PROGRAMMA 8 10 SINUSKROMMEN
110 PRINT CHR$(147)
120 HIRES 0,1
130 V=100:K=100:H=0.5:P=π/9:C=2*π/320
140 FOR N=0 TO 9
150 :  FOR J=0 TO 320 STEP 5
160 :     X=J*C:Y=INT(V-K*SIN(X+N*P)+H)
170 :     IF J=0 THEN X1=J:Y1=Y:GOTO 210
180 :     X2=J : Y2=Y
190 :     LINE X1,Y1,X2,Y2,1
200 :     X1=X2 : Y1=Y2
210 :  NEXT J
220 NEXT N
230 GET A$ : IF A$="" THEN 230
240 END
```

Programma 9 tekent een stelsel parabolen met de vergelijking

$y = -tx^2 + t$

Alle parabolen (bepaalde t-waarden) hebben dezelfde snijpunten met de x-as)X = 1 en x = -1).



```
100 REM PROGRAMMA 9 PARABOOLSTELSEL
110 PRINT CHR$(147)
120 HIRES 0,1
130 U=160:V=100:H=0.5
140 FOR K=-100 TO 100 STEP 10
150 :   FOR X=-110 TO 110 STEP 5
160 :      XX=INT(U+X+H)
170 :      Y=-K*X*X/6400+K:Y=INT(V-Y+H)
180 :      IF X=-110 THEN X1=XX:Y1=Y:GOTO220
190 :      X2=XX:Y2=Y
200 :      LINE X1,Y1,X2,Y2,1
210 :      X1=X2:Y1=Y2
220 :    NEXT X
230 NEXT K
240 GET A$ : IF A$="" THEN 240
250 END
```

Soms willen we de oppervlakte tussen de grafiek van een functie en de x-as berekenen (de integraal bepalen) of laten tekenen.

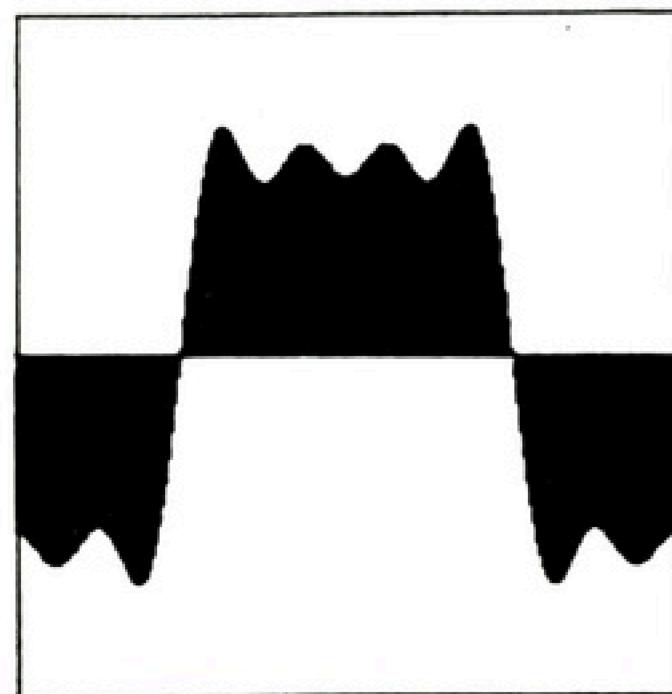
Programma 10 kleurt zo'n oppervlak tussen de x-as en de grafiek van de functie

$$y = \cos(x) - \frac{\cos(3x)}{3} + \frac{\cos(5x)}{5} - \frac{\cos(7x)}{7}.$$

Ook nu is de lusvariabele J de HRG-coördinaat X2. De waarde voor a is $-\pi$ en die voor b is $+\pi$, waaruit volgt dat

$$x = \frac{J.2\pi}{320} - \pi.$$

Ook nu nemen we in het programma $C = \frac{2\pi}{320}$



```
100 REM PROGRAMMA 10 OPPERVLAK ONDER EEN KROMME
110 PRINT CHR$(147)
120 HIRES 0,1
130 V=100:H=0.5:K=80:C=2*π/320
140 FOR J=0 TO 320
150 :  X=J*C-π : GOSUB 1000
160 :  Y=INT(V-K*Y+H)
170 :  LINE J,V,J,Y,1
180 NEXT J
190 GET A$: IF A$="" THEN 190
200 END
210 :
1000 Y=COS(X)-COS(3*X)/3+COS(5*X)/5-COS(7*X)/7
1010 RETURN
```

Probeer dit programma zo te maken dat de drie oppervlakken in bijvoorbeeld rood, wit en blauw gekleurd worden.

Niet-overal-continue functies

De functie $y = \frac{x^2+3}{x^2-x-6}$ kan niet met behulp van programma 7 getekend worden. Zouden we bijvoorbeeld voor x het interval $-5 \leq x \leq 5$ kiezen, dan liggen hierin de punten $x = -3$ en $x = 2$. Dit zijn de twee punten waarvoor de noemer van bovenstaande functie nul is. Als de computer bij het tekenen bij één van deze twee punten belandt, zal op het scherm een foutmelding (DIVISION BY ZERO) 'delen door nul' verschijnen en zal het programma afgebroken worden.

Dit zal ook gebeuren bij logaritmische functies met een niet-positief argument of bij wortelfuncties met een negatief argument. Zelfs bij het tekenen van continue (nette) functies kunnen problemen optreden. We komen hierbij in de problemen als de functiewaarden heel groot of heel klein worden. Transformatie van dergelijke waarden naar HRG-beeldschermcoördinaten (0-320 en 0-200) heeft dan geen enkele zin meer, omdat de aard van het verloop van de functie in het geheel niet meer tot uitdrukking komt. Het is daarom beter om de functiewaarden van een continue functie naar boven en beneden te begrenzen. Dit heeft tot gevolg dat de grafiek van een continue functie er uit zou kunnen zien als de grafiek van een niet-continue functie.

Het zal duidelijk zijn dat je een algemeen programma voor het tekenen van een willekeurige continue of niet-continue functie niet zomaar even opschrijft. Dergelijke programma's kom je in de vakliteratuur dan ook niet of nauwelijks tegen, en mocht je er wel een tegenkomen, dan betreft het òf een heel groot programma òf een programma dat voor een bepaalde functie, waarvan men van te voren weet waar de discontinuïteiten zitten, geschreven is.

We geven nu een verbazend kort programma voor het tekenen van de grafiek van een willekeurige functie $y = f(x)$. Veel wiskundeleraren, scholieren en studenten zullen nu hun 'oren' spitsen. U begrijpt dat, gezien het bovenstaande, hierbij wel enkele beperkingen gelden:

1. Degene die het programma gaat gebruiken moet een beetje kunnen programmeren. De functie moet namelijk in gedeelten in een subroutine (vanaf regel 1000) beschreven worden.
2. De programmagebruiker moet voldoende wiskundige kennis bezitten om te kunnen bepalen voor welke x-waarden functies moeilijkheden kunnen opleveren.

We zullen zien dat, normaal gesproken, slechts een paar BASIC-regels nodig zijn om de functiebeschrijving te programmeren. Het berekenen van nulpunten met behulp van een of ander numeriek-wiskundig algoritme is in het geheel niet nodig.

In het onderstaande programma 11 gebruiken we twee variabelen FZ en FA als zogeheten vlaggen (Flags). Een vlag is een variabele die slechts twee waarden (vaak 0 en 1) kan aannemen.

Om de werking van de vlag FZ duidelijk te maken geven we hieronder de subroutine 1000 met de functiebeschrijving van de functie

$$y = \frac{x^2+3}{x^2-x-6}$$

```
1000 N=X*X-X-6 : IF N=0 THEN FZ=1:RETURN
1010 Y=(X*X+3)/N
1100 IF Y<LP OR Y>HP THEN FZ=1:RETURN
1110 FZ=0:RETURN
```

De vlag FZ wordt alleen op 1 gezet als het punt (x,y) niet op het scherm getekend kan worden. Dit is het geval als de functie niet-continu is in het punt (x,y) (N = 0; x = -3 en x = 2) of als de functiewaarde buiten het opgegeven functiebereik ($LP \leqq y \leqq HP$) valt.

Waarvoor dient nu de tweede vlag FA? Bekijk het volgende stukje programma eens:

```
210 FA=1
220 FOR X=A TO B STEP DX
230 :   X2=INT(KX*(X-A)+H) : GOSUB 1000
240 :   IF FZ=1 THEN FA=1: GOTO 300
250 :   IF FA=1 THEN GOTO 290
260 :   Y2=INT(KY*(HP-Y)+H)
270 :   LINE X1,Y1,X2,Y2,1
280 :   X1=X2 : Y1=Y2 : GOTO 300
290 :   X1=X2:Y1=INT(KY*(HP-Y)+H) : FA=0
300 NEXT X
```

Als FZ = 1 dan wordt FA ook gelijk aan 1 gemaakt en wordt de volgende X-waarde bekeken (FOR-lus) zonder dat het nieuw berekende punt met het laatstgetekende punt verbonden wordt. Is FZ echter 0 (berekende punt ligt in het beeldvlak) dan wordt onderzocht of FA daarvoor soms op 1 gezet is. Is dit zo (tweede THEN) dan moet het nieuw berekende punt als nieuw beginpunt (X1,Y1) gekozen worden en dit mag dan ook niet met het vorige getekende punt verbonden worden. FA wordt in dit geval nul gemaakt.

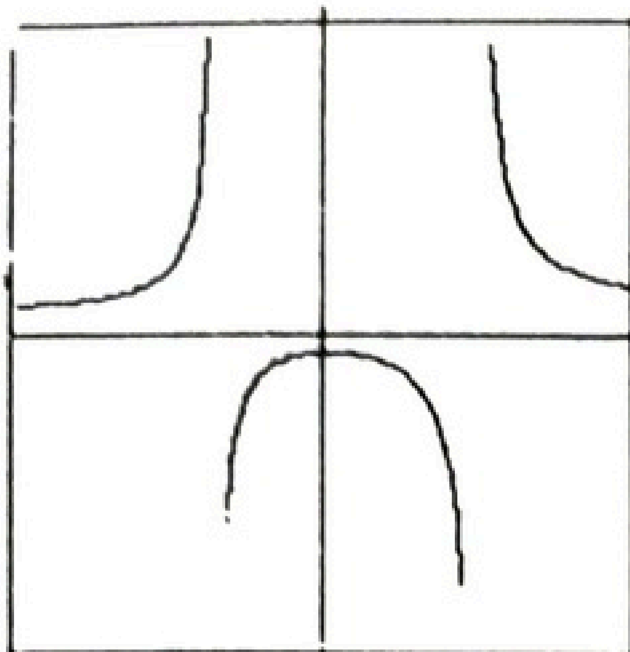
Dus als zowel FZ als FA nul zijn, wordt het nieuw berekende punt verbonden met het laatstgetekende punt.

Om te zorgen dat het programma probleemloos werkt moet voor het berekenen van het eerste punt van de grafiek, dus voor de FOR-lus, de vlag FA op 1 gezet worden, waardoor we er zeker van zijn dat de waarden X1 en Y1 berekend worden. Hier volgt het volledige programma:

```
100 REM PROGRAMMA 11 GRAFIEK VAN EEN WILLEKEURIGE FUNCTIE
110 PRINT CHR$(147)
120 INPUT "LINKERGRENS VOOR X "; A
130 INPUT "RECHTERGRENS VOOR X"; B
140 INPUT "BOVENGRENS VOOR Y  "; HP
150 INPUT "ONDERGRENS VOOR Y  "; LP
160 IF A>B THEN C=A:A=B:B=C
170 KX=320/(B-A):KY=200/(HP-LP):H=0.5
180 DX=(B-A)/256
190 PRINT CHR$(147)
200 HIRES 0,1
210 FA=1
220 FOR X=A TO B STEP DX
230 :   X2=INT(KX*(X-A)+H) : GOSUB 1000
240 :   IF FZ=1 THEN FA=1: GOTO 300
250 :   IF FA=1 THEN GOTO 290
260 :   Y2=INT(KY*(HP-Y)+H)
270 :   LINE X1,Y1,X2,Y2,1
280 :   X1=X2 : Y1=Y2 : GOTO 300
290 :   X1=X2:Y1=INT(KY*(HP-Y)+H) : FA=0
300 NEXT X
310 X1=0:Y1=INT(KY*HP+H):X2=320:Y2=Y1
320 IF Y1>0 AND Y1<=200 THEN LINE X1,Y1,X2,Y2,1
330 X1=INT(KX*(-A)+H):Y1=0:X2=X1:Y2=200
340 IF X1>0 AND X1<=320 THEN LINE X1,Y1,X2,Y2,1
350 GET A$ : IF A$="" THEN 350
360 END
1000 N=X*X-X-6 : IF N=0 THEN FZ=1:RETURN
1010 Y=(X*X+3)/N
1100 IF Y<LP OR Y>HP THEN FZ=1:RETURN
1110 FZ=0:RETURN
```

Dit programma tekent de grafiek van de functie

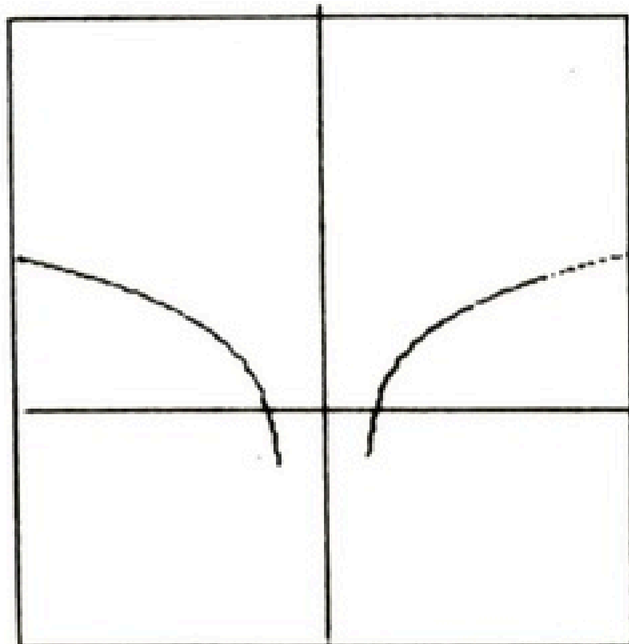
$y = \frac{x2+3}{x^2-x-6}$; kies voor A, B, HP en LP resp. -5, 5, 10 en -10



Wilt u een andere functie tekenen, bijvoorbeeld y = $\ln(x^2-2)$, herschrijf dan subroutine 1000 als volgt:

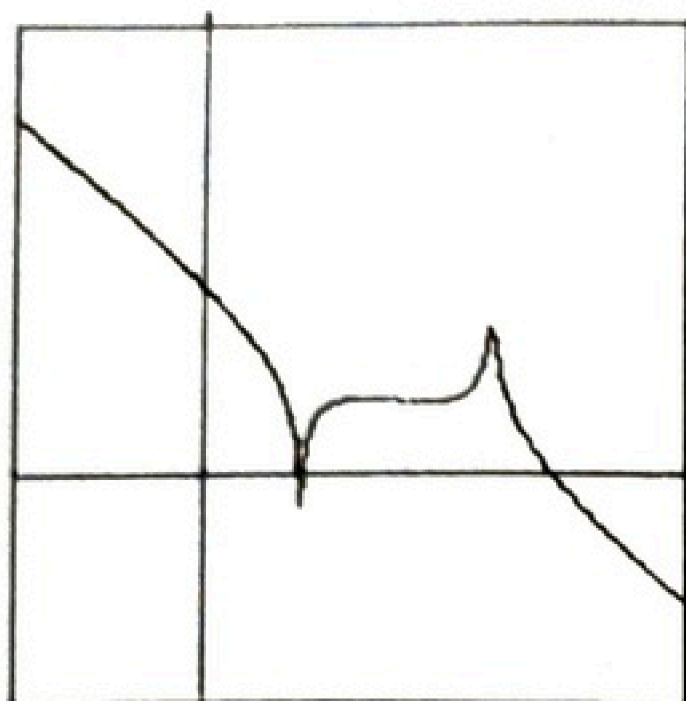
```
1000 U=X*X-2 : IF U<=0 THEN FZ=1:RETURN
1010 Y=LOG(U)
```

U krijgt dan deze grafiek:



Kies voor A, B, HP en LP de waarden -25, 25, 8 en -5.

Hier volgt een hele fraaie:



$$y=3-x+\ln\left|\frac{x-1}{x-3}\right|$$

Subroutine 1000 wordt nu:

```
1000 N=X-3 : IF N=0 THEN FZ=1:RETURN
1010 Y=3-X+LOG(ABS((X-1)/N))
```

Kies voor A, B, HP en LP de waarden -5, 10, 8 en -4.

3 Krommen in poolcoördinaten en in parametervorm

In het vorige hoofdstuk hebben we ons beziggehouden met het tekenen van de grafiek van een continue of niet-continue functie, waarvan de vergelijking als $y = f(x)$ geschreven kon worden; dus in cartesische vorm. Nu gaan we grafieken tekenen van functies waarvan de vergelijking in poolcoördinaten geschreven wordt of in de zogeheten parametervorm. Dit levert zeer fraaie 'beelden' op.

Krommen met poolcoördinaten

Functies, waarvan de grafiek een gesloten kromme laat zien, zijn vaak eenvoudiger met poolcoördinaten te beschrijven dan in cartesische vorm. Als voorbeeld noemen we de 'hartkromme' (kardioide), waarvan de vergelijking in poolcoördinaten eenvoudig is, namelijk:

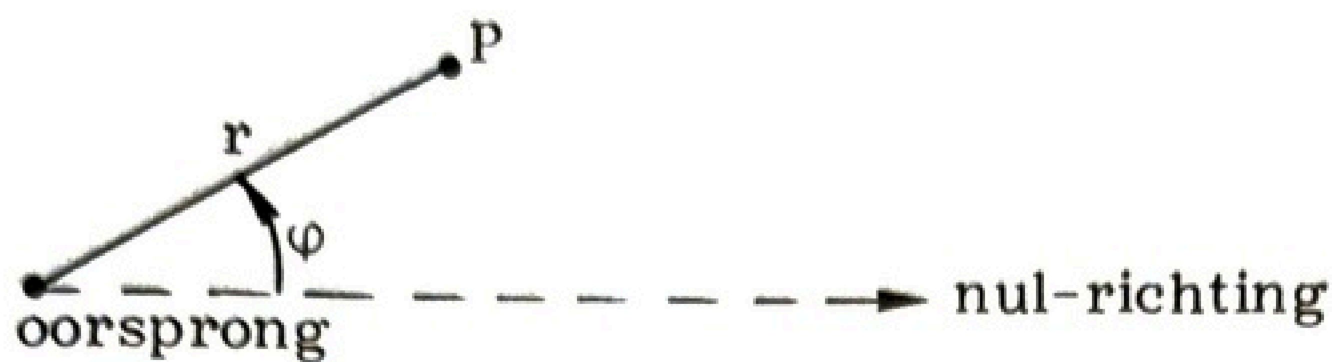
$$r = k(1 + \cos\varphi),$$

maar waarvan de cartesische vorm nogal ingewikkeld is, namelijk

$$\frac{(x^2-y^2-kx)^2}{k^2(x^2+y^2)} = 1.$$

Om de volgende programma's te doorgronden (niet om ze te draaien!) is een beetje theorie nodig. Het poolcoördinatenstelsel wordt in de wiskundelessen op school niet of nauwelijks behandeld. Eigenlijk is dit jammer, want hiermee (en met de parametervorm) kunnen juist de mooiste functies (en daarmee de fraaiste grafieken) beschreven en getekend worden.

In een poolcoördinatenstelsel wordt elk punt in het platte vlak met twee coördinaten, te weten r en φ, bepaald. r is de afstand tussen het punt en de oorsprong en φ (phi, spreek uit: fie) is de hoek tussen de horizontale lijn door de oorsprong en de lijn door de oorsprong en het punt (zie tekening op p.31).

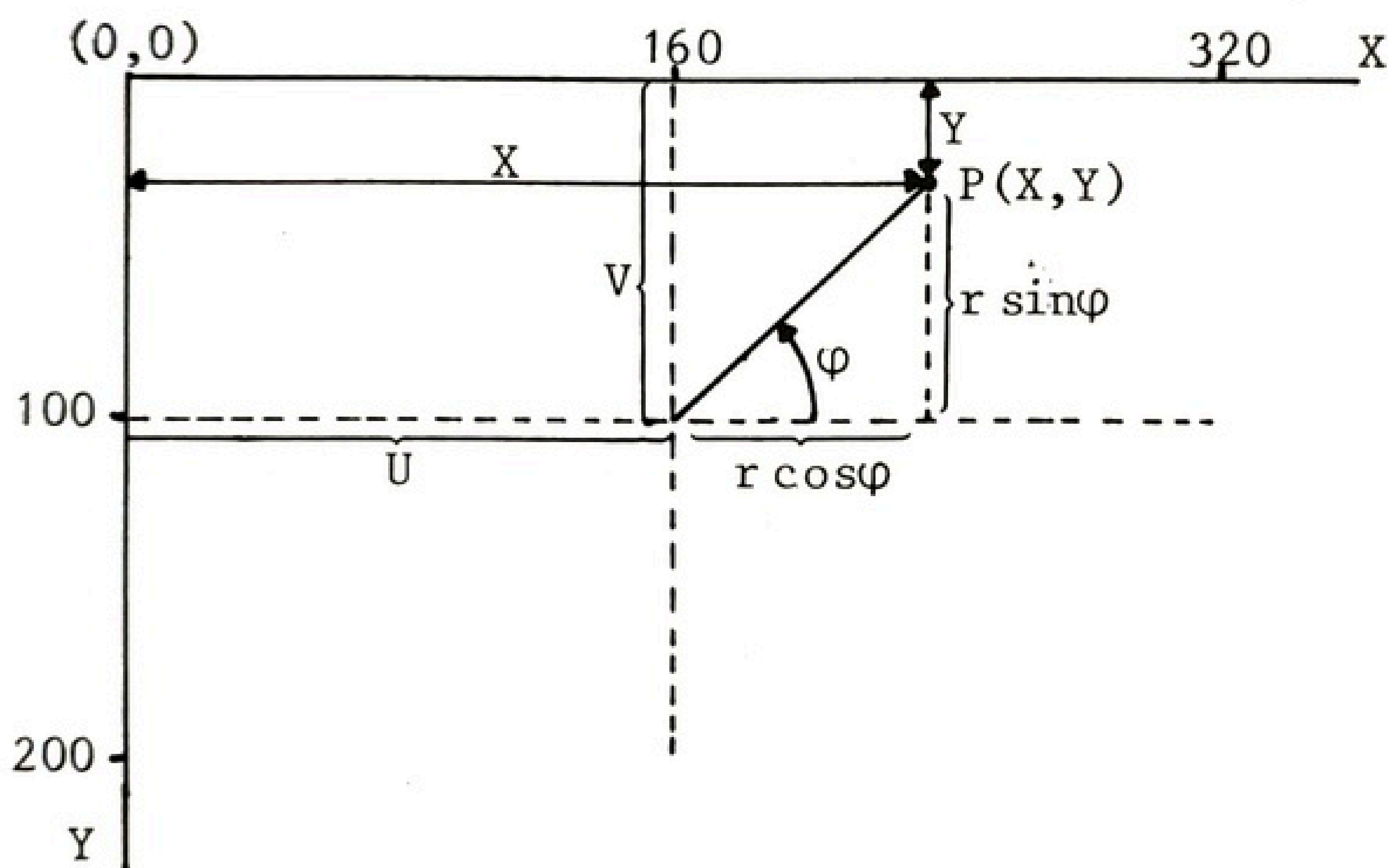


De oorsprong van het poolcoördinatenstelsel leggen we in het middelpunt van het beeldscherm (U = 160, v = 100). Zoals we weten ligt de oorsprong van het HRG-beeldschermcoördinatenstelsel in de linkerbovenhoek van het beeldscherm (HOME-positie). De nul-richting in ons poolcoördinatenstelsel is horizontaal en wijst naar rechts. φ=90° (π/2 radialen) is verticaal naar boven; φ=180° (π radialen) is horizontaal naar links en φ=270° (3π radialen) verticaal naar beneden.

We gebruiken de volgende twee transformatieformules om het, uit de functievergelijking berekende, punt P(r,φ) om te zetten in het beeldschermpunt P(X,Y):

```
X = INT(U + R*COS(P) + 0.5)     èn
Y = INT(V - R*SIN(P) + 0.5)
```

In onderstaande figuur zien we hiervoor de verklaring.



De hoek φ, in de programma's voorgesteld door de variabele P, doorloopt steeds in positieve richting (tegen de klok in) het interval van 0 tot 2π. Dit is een interval in radialen (0-360 in graden).

In alle programma's is als lusvariabele de hoek W in graden gekozen. Met de formule P = W.(π/180) (in de programma's P = W*RD) wordt de hoek in radialen berekend. Het voordeel van een lusvariabele

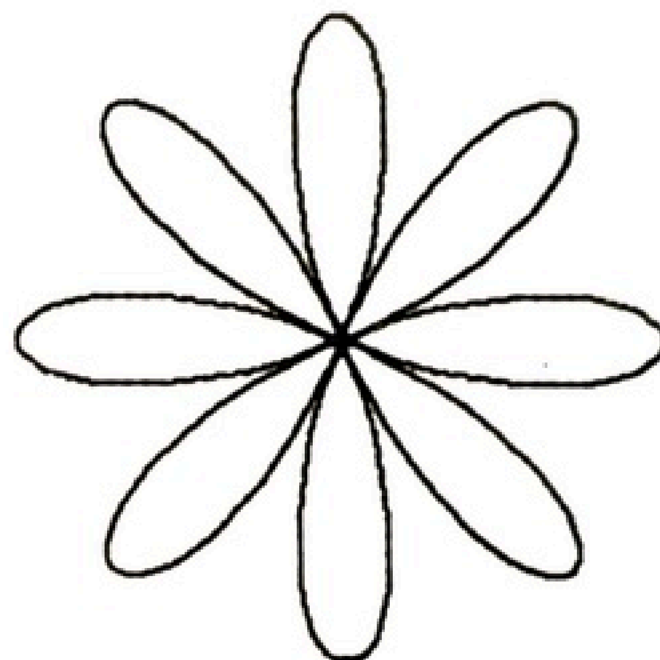
die het interval 0-360° doorloopt is dat de stapgrootte waarmee de lusvariabele steeds wordt opgehoogd een geheel getal kan zijn (1 bijvoorbeeld) en dat hierdoor de programma's beter leesbaar zijn. Een neveneffect is dat alle programma's bijna woordelijk in Pascal zijn over te zetten; Pascal kent immers geen gebroken stapgrootte in een lus.

Meer theorie hebben we niet nodig. De volgende programma's spreken grotendeels voor zichzelf.

Programma 12 tekent de grafiek van de functie

$$r = k\cos(n\varphi).$$

In het programma kiezen we k = 110 (K) en n = 4 (N).



```
100 REM PROGRAMMA 12 GRAFIEK VAN DE
        FUNKTIE R=100*COS(4*PHI)
110 PRINT CHR$(147)
120 HIRES 0,1
130 U=160:V=100:H=0.5:RD=π/180:K=100
140 FOR W = 0 TO 360 STEP 3
150 :  P=W*RD : GOSUB 1000
160 :  X = INT(U+K*R*COS(P)+H)
170 :  Y = INT(V-K*R*SIN(P)+H)
175 :  IF P=0 THEN X1=X:Y1=Y : GOTO 210
180 :  X2=X : Y2=Y
190 :  LINE X1,Y1,X2,Y2,1
200 :  X1=X2 : Y1=Y2
210 NEXT W
220 GET A$ : IF A$="" THEN 220
230 END
1000 :  R=COS(4*P) : R=ABS(R)
1010 RETURN
```

Het programma tekent een achtdelige draaisymmetrische figuur; een bloem met acht blaadjes. Als u met dit programma gaat experimenteren zult u snel ontdekken dat voor even waarden van n een 2n-bladerige en voor oneven waarden van n een n-bladerige bloem ontstaat. Kunt u een tweekleurige bloem maken?

Wilt u dat uw computer sneller tekent? Neem dan een stapgrootte van 3° of van 5° in plaats van 1°. Bedenk dan wel dat de blaadjes wat hoekiger worden.

Als u regel 1000 R = COS(4*P) vervangt door één van de onderstaande functies, krijgt u steeds een andere mooie figuur:

```
R = COS(4*SIN(2*P)) : R = ABS(R)
R = COS(4*SIN(3*P)) : R = ABS(R)
R = SIN(3*SIN(2*P))  : R = ABS(R)
R = SIN(5*COS(2*P)) : R = ABS(R)
```

U kunt naar hartelust trigonometrische functies met verschillende parameters 'mengen'. Laat uw creativiteit en nieuwsgierigheid de vrije loop.

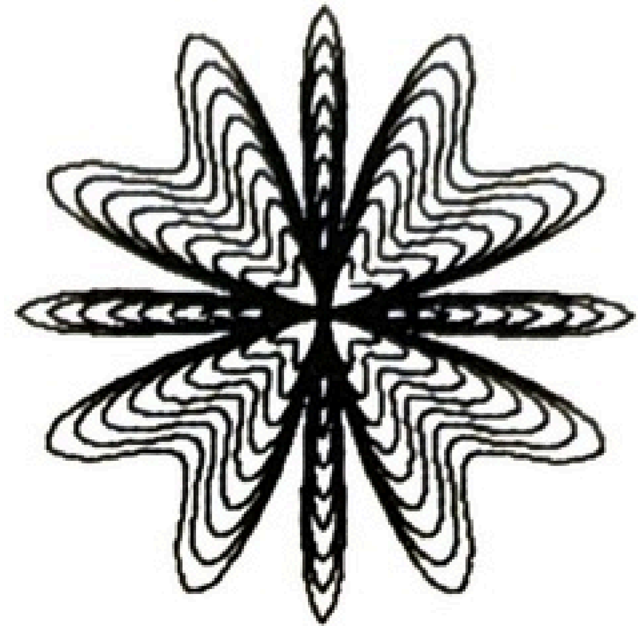
Omdat de poolcoördinaat r per definitie niet negatief mag zijn, moeten we in regel 1000 ook de opdracht R = ABS(R) opnemen.

Nog veel mooiere figuren krijgen we als we niet één grafiek, maar een stelsel van gelijkvormige grafieken tekenen.

Programma 13 tekent het stelsel krommen:

R = K*COS(4*SIN(2*P)).

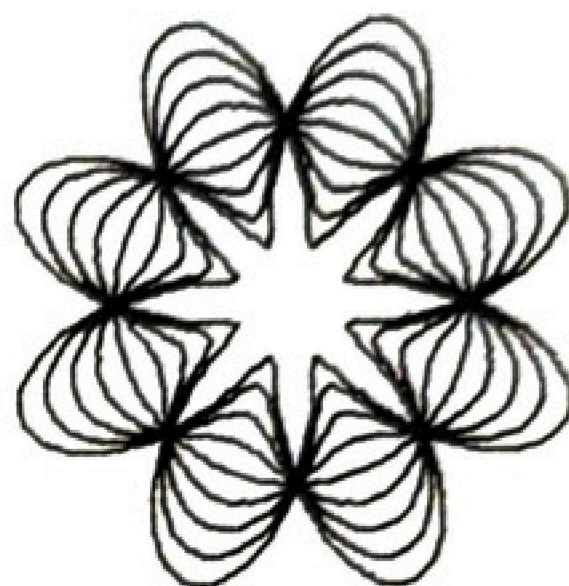
De parameter K loopt van 20 tot 100 in stappen van 10.



```
100 REM PROGRAMMA 13 GRAFIEK VAN DE
        FUNKTIE R=K*COS(4*SIN(2*P))
110 PRINT CHR$(147)
120 HIRES 0,1
130 U=160:V=100:H=0.5:RD=π/180
140 FOR K=20 TO 100 STEP 10
150 :   FOR W=0 TO 360 STEP 2
160 :      P=W*RD : GOSUB 1000
170 :      X=INT(U+K*R*COS(P)+H)
175 :      IF P=0 THEN X1=X:Y1=Y : GOTO 2
10
180 :      Y=INT(V-K*R*SIN(P)+H)
190 :      IF P=0 THEN X1=X:Y1=Y:GOTO 210

200 :      X2=X : Y2=Y
210 :      LINE X1,Y1,X2,Y2,1
220 :      X1=X2:Y1=Y2
230 :   NEXT W
240 NEXT K
250 GET A$ : IF A$="" THEN 250
260 END
1000 R=COS(4*SIN(2*P)) : R=ABS(R)
1010 RETURN
```

Programma 14 tekent sinuskrommen die cirkelvormig gekromd zijn.



```
100 REM PROGRAMMA 14 SINUSKROMMEN IN CIRKELS
110 PRINT CHR$(147)
120 HIRES 0,1
130 U=160:V=100:H=0.5:RD=π/180
140 FOR K=-40 TO 40 STEP 10
150 :   FOR W=0 TO 360 STEP 2
160 :      P=W*RD : GOSUB 1000
170 :      X=INT(U+R*COS(P)+H)
180 :      Y=INT(V-R*SIN(P)+H)
190 :      IF P=0 THEN X1=X:Y1=Y:GOTO 230
200 :      X2=X : Y2=Y
210 :      LINE X1,Y1,X2,Y2,1
220 :      X1=X2:Y1=Y2
230 :   NEXT W
240 NEXT K
250 GET A$ : IF A$="" THEN 250
260 END
1000 R=60 + K*SIN(4*P)
1010 RETURN
```

Probeert u eens

```
1000 R = 2*TAN("*P)+K*SIN(2*COS(SIN(6*P)))
```

voor een schitterend spinneweb. De variaties zijn echt onbegrensd!

Programma 15 tekent een bloemvormige figuur met blaadjes die in het midden aan steeltjes vastzitten. In het programma is N = 4 gekozen. U kunt gerust andere waarden voor N proberen. Het effect is hetzelfde als bij programma 12.



```
100 REM PROGRAMMA 15 BLOEMEN
110 PRINT CHR$(147)
120 HIRES 0,1
130 U=160:V=100:H=0.5: RD=π/180
140 N=4: C=0.25:REM EERST DE BLOEMEN
150 FOR K=30 TO 80 STEP 10
160 :   FOR W=0 TO 360 STEP 3
170 :       P=W*RD:R=K*(1+C*ABS(SIN(N*P)))
180 :       X=INT(U+R*COS(P)+H)
190 :       Y=INT(V-R*SIN(P)+H)
200 :       IF P=0 THEN X1=X:Y1=Y:GOTO 240
210 :       X2=X : Y2=Y
220 :       LINE X1,Y1,X2,Y2,1
230 :       X1=X2: Y1=Y2
240 :   NEXT W
250 NEXT K
260 R=30:P1=(180/N)*RD:REM DAN DE STELEN
270 FOR J=1 TO N
280 :   P=J*P1
290 :   X1=INT(U+R*COS(P)+H)
300 :   Y1=INT(V-R*SIN(P)+H)
310 :   X2=INT(U+R*COS(P+π)+H)
320 :   Y2=INT(V-R*SIN(P+π)+H)
330 :   LINE X1,Y1,X2,Y2,1
340 NEXT J
350 GET A$ : IF A$="" THEN 350
360 END
```

Bezit u een kleurenmonitor of een kleurenplotter dan kunt u na elke doorloop van de K-lus de afdrukkleur veranderen. Ook de stelen van de bloem kunnen een eigen kleur krijgen. Tekent u een aantal van dergelijke bloemen naast of onder elkaar dan hebt u een begin gemaakt met 'computer-art'.

Spiralen zijn altijd geliefde figuren. Programma 16 tekent logaritmische spiralen of spiralen van Archimedes. Deze laatste soort spiralen hebben als vergelijking

$$r = c.\varphi$$

In het programma op p.40 is voor C de waarde 3 gekozen. Kies gerust andere waarden, maar wel tussen 0.5 en 20.

De logaritmische spiraal heeft (naar Bernoulli) de vergelijking

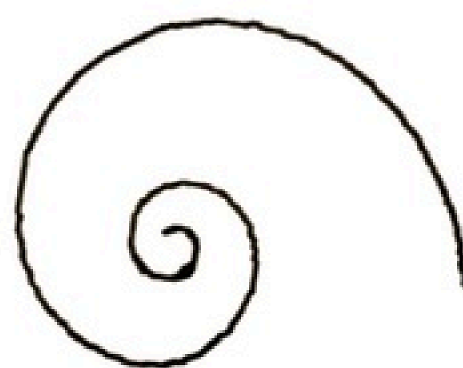
$$r = k\,e^{c\varphi}.$$

In het programma dat de onderste spiraal getekend heeft hebben we voor K de waarde 110 en voor C de waarde -0.2 gekozen.



$r=2\varphi$

$r=110e^{-0,2\varphi}$



Omdat de logaritmische spiraal zich oneindig vaak rond de oorsprong zal winden moet ervoor gezorgd worden dat het programma na bepaalde 'tijd' wordt afgebroken.

```
100 REM PROGRAMMA 16 SPIRALEN
110 PRINT CHR$(147)
120 HIRES 0,1
130 U=160:V=100:H=0.5:RD=π/180
140 C=3
150 FOR W=0 TO 10000 STEP 3
160 :   P=W*RD : GOSUB 1000
170 :   X=INT(U+R*COS(P)+H)
180 :   Y=INT(V-R*SIN(P)+H)
190 :   IF P=0 THEN X1=X:Y1=Y:GOTO 240
200 :   IF X<0 OR X>320 OR Y<0 OR Y>200 THEN 250
210 :   X2=X : Y2=Y
220 :   LINEX1,Y1,X2,Y2,1
230 :   X1=X2: Y1=Y2
240 NEXT W
250 GET A$ : IF A$="" THEN 250
260 :
1000 R=C*P
1010 RETURN
```

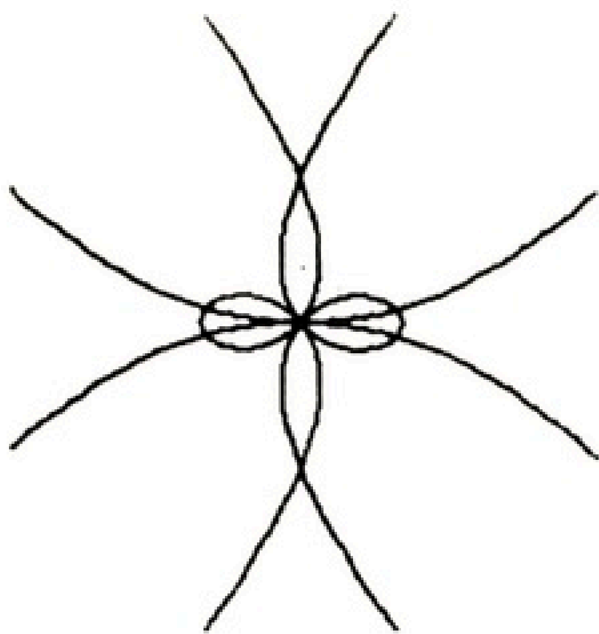
Voeg toe regel 245 U = 165 : GOTO 150 en zie hoe dit de figuur verfraait.

Voor het tekenen van een logaritmische spiraal maken we C gelijk aan -0.2 en herschrijven subroutine 1000 als volgt:

```
1000 R=110*EXP(C*P)
1010 IF R<5 THEN END
1020 RETURN
```

Als laatste programma met poolcoördinaten geven we het programma 17 voor het tekenen van een willekeurige, in poolcoördinaten geformuleerde, continue of niet-continue functie. De programmastructuur komt overeen met de structuur van programma 11 (zie p.27). U kunt daar de werking van de vlaggen FZ en FA nog eens bestuderen. Ook de andere variabelen hebben dezelfde betekenis als hun naamgenoten in programma 11. Als voorbeeld in het programma hebben we een vrij ingewikkelde functie, die niet overal continu is, gekozen:

$$r = \frac{\sin(1{,}5.\varphi)}{1-2.\cos\varphi}.$$



De figuur is getekend met A = -2, B = 2, LP = -2, HP = 2, WO = 0° en WN = 720°. Om de grafiek er optisch mooi uit te laten zien is de regel

```
1090 IF R<0 THEN F1=1 : RETURN
```

weggelaten.

Wilt u een andere functie proberen, verander dan alleen iets in de regels 1000 t/m 1080. Verander de regels 1090 t/m 1120 niet.

```
100 REM PROGRAMMA 17 GRAFIEK VAN DE FUNCTIE R=F(PHI)
110 PRINT CHR$(147)
120 INPUT "LINKERGRENS VOOR X  "; A
130 INPUT "RECHTERGRENS VOOR X "; B
140 INPUT "ONDERGRENS VOOR Y   "; LP
150 INPUT "BOVENGRENS VOOR Y   "; HP
160 INPUT "STARTWAARDE VOOR PHI"; W0
170 INPUT "EINDWAARDE VOOR PHI"; WN
180 KX=320/(B-A):KY=200/(HP-LP):H=0.5:RD=π/180
190 PRINT CHR$(147)
200 HIRES 0,1
210 FA=1
220 FOR W=W0 TO WN
230 :  P=W*RD : GOSUB 1000
240 :  IF FZ=1 THEN FA=1 : GOTO 320
250 :  IF FA=1 THEN GOTO 290
260 :  X2=INT(KX*(X-A)+H)
270 :  Y2=INT(KY*(HP-Y)+H)
280 :  LINE X1,Y1,X2,Y2,1:X1=X2:Y1=Y2:GOTO 320
290 :  X1=INT(KX*(X-A)+H)
300 :  Y1=INT(KY*(HP-Y)+H)
310 FA=0
320 NEXT W
330 GET A$ : IF A$="" THEN 330
340 END
1000 N=1-2*COS(P) : IF N=0 THEN FZ=1:RETURN
1010 R=SIN(3*P/2)/N
1100 X=R*COS(P) : Y=R*SIN(P)
1110 IF X<A OR X>B OR Y<LP OR Y>HP THENFZ=1:RETURN
1200 FZ=0:RETURN
```

De parametervorm

De meest interessante krommen krijgen we bij functies waarvan de vergelijking in parametervorm wordt opgesteld. Dit houdt in dat zowel de X- als de Y-coördinaat als functie van dezelfde, derde, parameter t worden uitgedrukt. In de natuurkunde zien we vaak de tijd als parameter (vandaar de t). In de wiskunde is de parameter t bijna altijd een hoek in radialen.

Zo is de parametervorm van een cirkel met straal r en het middelpunt in de oorsprong:

$$x = r\cos t \qquad \text{en} \qquad y = r\sin t$$

Uit deze twee vergelijkingen kan gemakkelijk de cartesische vorm $x^2+y^2=r^2$ gedistilleerd worden. De vergelijking van een ellips met het middelpunt in de oorsprong en met een halve lange as a en een halve korte as b is:

$$x = a\cos t \qquad \text{en} \qquad y = b\sin t$$

Het is eenvoudig programma's te ontwikkelen voor het tekenen van stelsels concentrische of excentrische cirkels. Ook stelsels ellipsen zijn, dankzij de parametervorm, eenvoudig te tekenen. We doen dit echter niet, omdat de figuren niet zo bijzonder zijn en vrij bekend. In hoofdstuk 1 hebben we trouwens al een ellips met behulp van de parametervorm geprogrammeerd.

De volgende programma's tekenen figuren die u mogelijk nog niet kent en die zich voor computer-graphics bijzonder goed lenen.

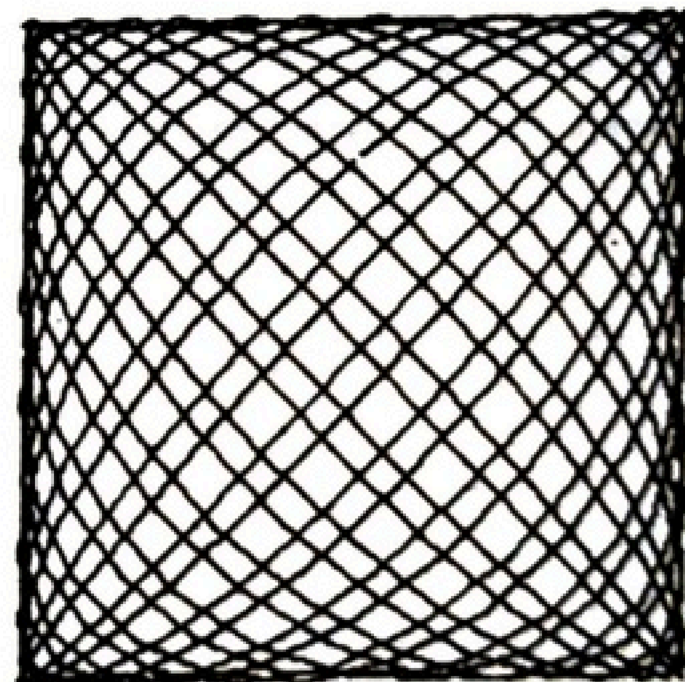
Programma 18 tekent een Lissajousfiguur. In het algemeen is de parametervorm voor een dergelijke figuur:

$$x = k_1.\sin(f_1.t+p_1) + k_2.\cos(f_2.t)$$
$$y = k_3.\sin(f_3.t+p_3) + k_4.\cos(f_4.t)$$

Hierin is p de fase, f de frequentie en k de amplitude. Om een mooie figuur te krijgen hebben we de volgende waarden gekozen:

k_1=k_3=100, f_1=16, p_1=0, f_3=17, p_3=35 en k_2=k_4=f_2=f_4=0

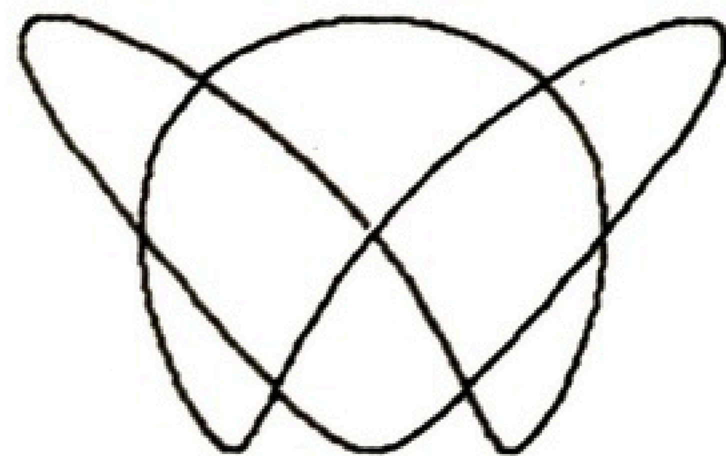
```
100 REM PROGRAMMA 18 LISSAJOUSFIGUREN
110 PRINT CHR$(147)
120 PRINT "TOETS K1,F1,P1,K2,F2 IN";
130 INPUT K1,F1,P1,K2,F2
140 PRINT:PRINT
150 PRINT "TOETS K3,F3,P3,K4,F4 IN";
160 INPUT K3,F3,P3,K4,F4
170 U=160:V=100:H=0.5:RD=π/180
180 PRINT CHR$(147)
190 HIRES 0,1
200 FOR W=0 TO 360
210 :   T=W*RD : GOSUB 1000
220 :   X=INT(U+X+H)
230 :   Y=INT(V-Y+H)
240 :   IF W=0 THEN X1=X:Y1=Y:GOTO 280
250 :   X2=X : Y2=Y
260 :   LINE X1,Y1,X2,Y2,1
270 :   X1=X2: Y1=Y2
280 NEXT W
290 GET A$ : IF A$="" THEN 290
300 END
310 :
1000 X=K1*SIN(F1*T+P1)+K2*COS(F2*T)
1010 Y=K3*SIN(F3*T+P3)+K4*COS(F4*T)
1020 RETURN
```



Experimenteer met het programma! Zoek zelf de juiste waarden voor de constanten zodat u mooie figuren krijgt. Natuurkundigen zullen u hierbij graag helpen; zij weten welke waarden u moet nemen!

Programma 19 presenteert een niet alledaagse figuur. Deze figuur wordt dikwijls de 'vliegenkopfiguur' genoemd. Om de vliegenkop horizontaal op het beeldscherm te krijgen moet de parameter t het interval 90°-450° doorlopen.

```
100 REM PROGRAMMA 19 VLIEGENKOP
110 PRINT CHR$(147)
120 HIRES 0,1
130 U=160:V=100:H=0.5:K=30:RD=π/180
140 FOR W=90 TO 450 STEP 3
150 :   T=W*RD : GOSUB 1000
160 :   X=INT(U+X+H)
170 :   Y=INT(V-Y+H)
180 :   IF W=90 THEN X1=X:Y1=Y:GOTO 220
190 :   X2=X : Y2=Y
200 :   LINE X1,Y1,X2,Y2,1
210 :   X1=X2: Y1=Y2
220 NEXT W
230 GET A$: IF A$="" THEN 230
240 END
1000 X=K*SIN(2*T)*(2.5+COS(3*T))
1010 Y=K*2*COS(3*T)
1020 RETURN
```



Het tekenen van de x- en y-as is achterwege gelaten om de 'kop' beter te laten uitkomen.

Voor een echte vliegenkop met 'ogen' moet u toevoegen:

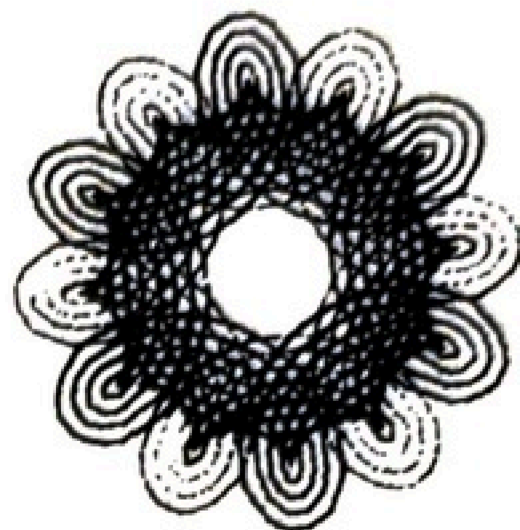
```
221 CIRCLE 120,100,10,10,1
222 CIRCLE 200,100,10,10,1
223 PAINT 105,100,1
224 PAINT 215,100,1
```

Hoe geraffineerder u de functies x = f(t) en y = f(t) kiest, des te ongewoner worden de figuren. Probeer hier hoe creatief u kunt zijn.

De 'huiswiskundigen' bij computerfabrikanten hebben vaak hun eigen lievelingsfuncties. Die zie je dan ook vaak in een demonstratieprogramma optreden.

Programma 20 tekent een stelsel krommen. Hewlett-Packard publiceerde deze tekening enige tijd geleden. Onder het programma staat een tabel met waarden voor A en B, die heel mooie figuren opleveren. Probeer zelf andere combinaties. U zult verrukt zijn over wat u ziet!

A=-6/B=1



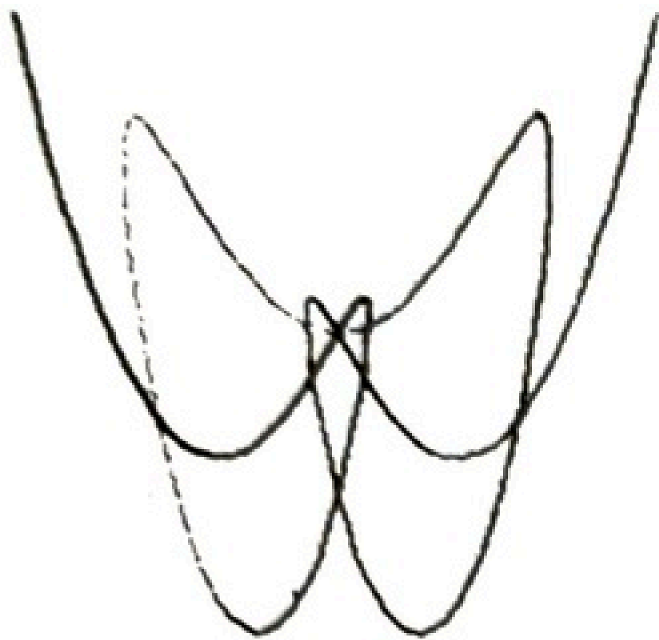
```
100 REM PROGRAMMA 20 VLINDERS
110 PRINT CHR$(147)
120 U=160:V=100:H=0.5:RD=π/180
130 KX=10 : KY=10
140 INPUT "TOETS A EN B IN"; A,B
150 PRINT CHR$(147)
160 HIRES 0,1
170 FOR N=-3 TO 3
180 :   FOR W=0 TO 360 STEP 3
190 :   T=W*RD
200 :   X=(A+B)*COS(T)-N*B*COS((A+B)/B*T)
210 :   Y=(A+B)*SIN(T)-N*B*SIN((A+B)/B*T)
220 :   XX=INT(U+KX*X+H)
230 :   YY=INT(V-KY*Y+H)
240 :   IF W=0 THEN X1=XX:Y1=YY:GOTO 280
250 :   X2=XX : Y2=YY
260 :   LINE X1,Y1,X2,Y2,1
270 :   X1=X2 : Y1=Y2
280 :   NEXT W
290 NEXT N
300 GET A$ : IF A$="" THEN 300
310 END
```

Tabel voor mooie figuren

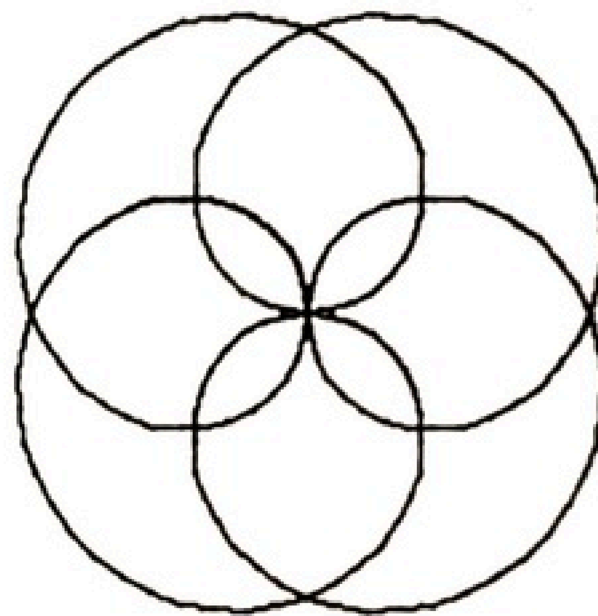
A	-6	-6	-8	4	4	6	4,5	....
B	1	2	2	1	2	1	1,5	....

Kies in deze programma's de stapgrootte voor hoek W (FOR W=.....) gerust 1; dit duurt wat langer maar geeft mooieren tekeningen.

Het laatste programma uit dit hoofdstuk tekent 'supersymmetrische' figuren. Ook nu geven we een tabel met waarden voor de parameters A, B en C. Het is ongelooflijk hoe de figuur totaal verandert als we andere waarden voor één of meer parameters kiezen. Het idee voor **programma 21** is afkomstig uit het Amerikaanse tijdschrift Creative Computing.



A=2/B=7/C=3



A=6/B=6/C=4

```
100 REM PROGRAMMA 21 SYMMETRISCHE KROMMEN
110 PRINT CHR$(147)
120 INPUT "TOETS A,B,C IN"; A,B,C
130 U=160:V=100:H=0.5:RD=π/180:K=100
140 PRINT CHR$(147)
150 HIRES 0,1
160 FOR W=0 TO 360
170 :   T=W*RD : R=K*SIN(C*T)
180 :   X2=INT(U+R*COS(A*T)+H)
190 :   Y2=INT(V-R*SIN(B*T)+H)
200 :   IF W=0 THEN X1=X2:Y1=Y2:GOTO 230
210 :   LINE X1,Y1,X2,Y2,1
220 :   X1=X2: Y1=Y2
230 NEXT W
240 GET A$ : IF A$="" THEN 240
250 END
```

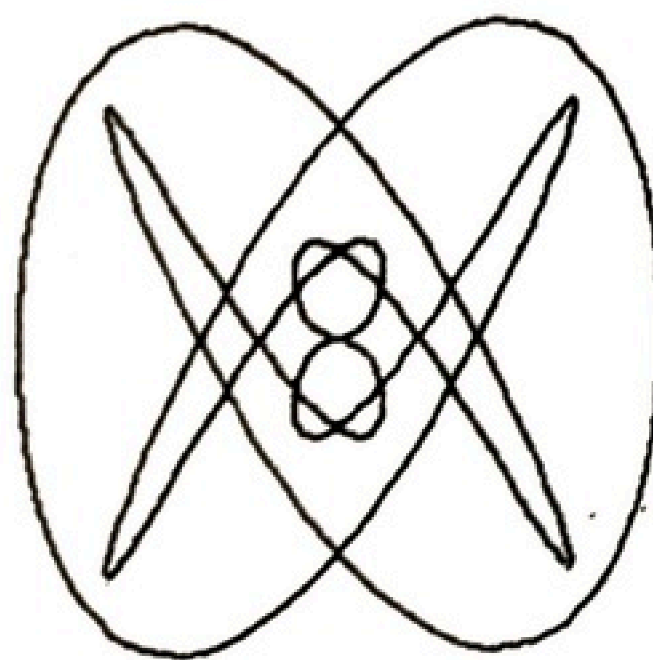
Tabel voor mooie figuren

A	2	6	4	1	3	2	....
B	7	6	6	1	3	2	....
C	3	4	1	4	5	9	....

Het zou jammer zijn als u bij het zien van een mooie figuur de waarden voor A, B en C niet hebt genoteerd. Verander het programma zo, dat na afloop van het tekenen de waarden van A, B en C op het beeldscherm of op de printer worden afgedrukt. Ook heel leuk zijn de combinaties:

A	20	-40	100
B	-1	-40	-0.5
C	3	10	3

A=4/B=6/C=1



4 *Tekenen van driedimensionale figuren*

In dit en het volgende hoofdstuk gaan we ons bezighouden met het tekenen van driedimensionale lichamen (zo heet dat in de wiskunde) zoals kubussen, prisma's, pyramiden, kegels en bollen, en met het tekenen van driedimensionale grafieken van functies. Zo'n driedimensionale grafiek is een vlak in de ruimte met een vergelijking van de vorm $z = f(x,y)$. Zo krijgen we de bekende 'Mexicaanse hoed' (zie p.73) als we de functie $z = e^{-(x^2+y^2)}$ tekenen (e is het grondtal van de natuurlijke logaritme; e = 2,71828...).

Er bestaan veel programma's waarmee driedimensionale figuren getekend kunnen worden. Waarom we toch ook in dit boek dergelijke programma's laten zien komt, omdat er aan de meeste van deze programma's drie nadelen kleven:

1. De meeste programma's voor driedimensionaal-tekenen zijn voor een bepaald graphics-systeem ontworpen en zijn slechts met veel inspanning geschikt te maken voor andere systemen.

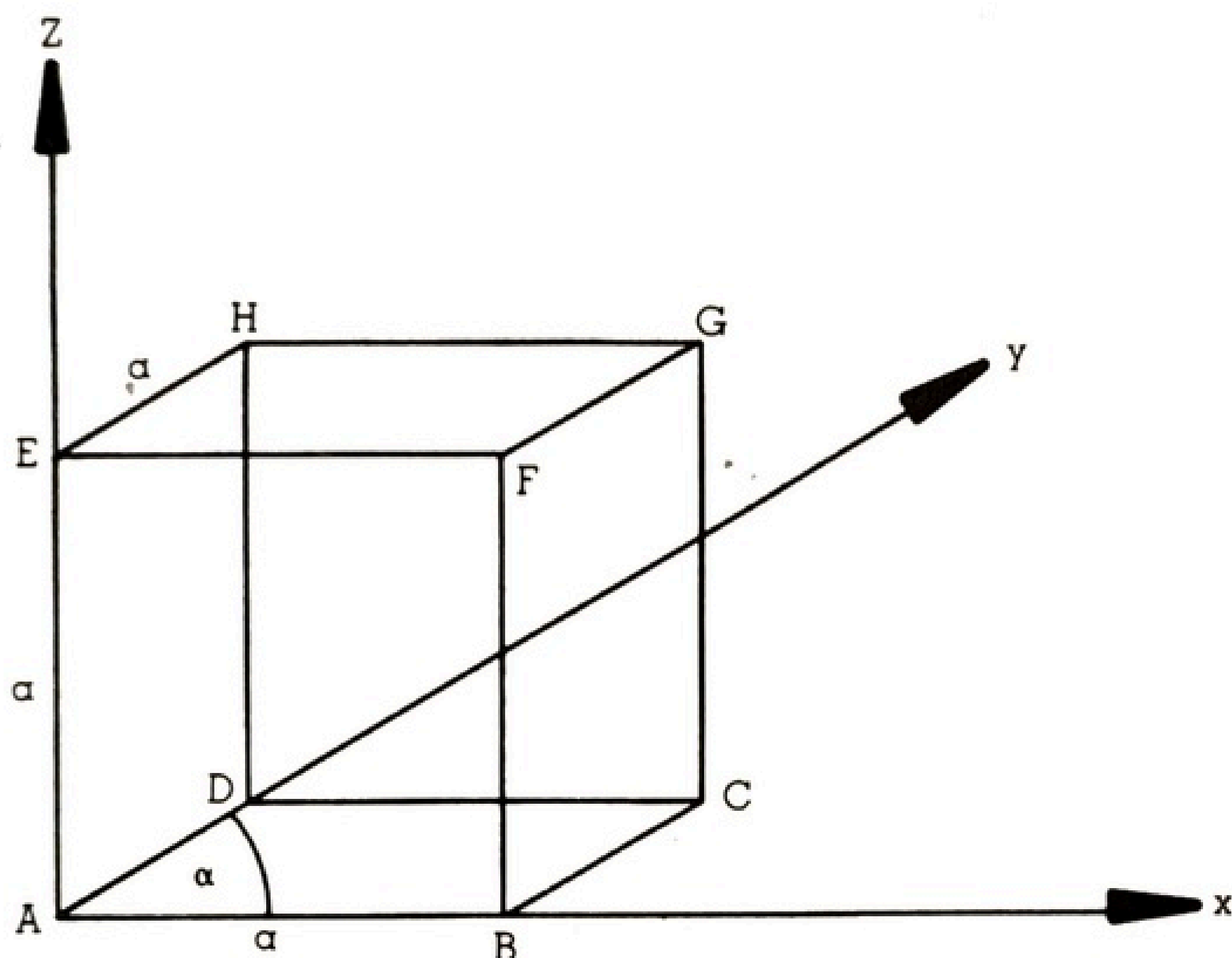
2. De wiskundige theorie die aan het driedimensionaal-tekenen ten grondslag ligt wordt zelden of zeer summier behandeld.

3. Veel programma's zijn uiterst ingenieus ontworpen en draaien op de kleine microcomputers, in BASIC, heel langzaam.

Met de volgende programma's en stukjes theorie hopen we deze drie nadelen uit de weg te ruimen.

Er bestaan verschillende methoden om een driedimensionaal lichaam, bijvoorbeeld een kubus, in een plat vlak te projecteren. Wij hanteren de projectiemethode, die u wellicht op school gebruikt hebt (of nog gebruikt). Stelt u zich een doorzichtige kubus voor met ribben van draadijzer. U staat voor deze kubus een beetje rechts van het

midden en kijkt er zo'n beetje schuin bovenop. Uw gezichtsstralen projecteren elke hoek, met de daaraan verbonden ribben, van de kubus in een, achter de kubus liggend, projectievlak. Zo ontstaat de bekende 'schuine' afbeelding van een kubus.



De verticale en horizontale ribben worden op ware grootte getekend. De ribben die naar achteren lopen worden verkort weergegeven.

a' = k.a   k = verkleiningsfactor

De rechte hoek tussen de ribben BA en AD in het grondvlak lijkt eveneens kleiner te zijn geworden (zie α in de bovenstaande figuur). U kunt gemakkelijk nagaan dat door het vastleggen van alpha (bijvoorbeeld 45°) en k (bijvoorbeeld 0.5) de projectierichting eenduidig bepaald is.

Wat we nodig hebben zijn transformatievergelijkingen die voor een bepaalde α en k de coördinaten x,y,z van een punt op de ruimtelijke kubus projecteren op de coördinaten x' en y' van het geprojecteerde punt in het platte (projectie-)vlak. Deze formules vormen de kern van alle volgende programma's.

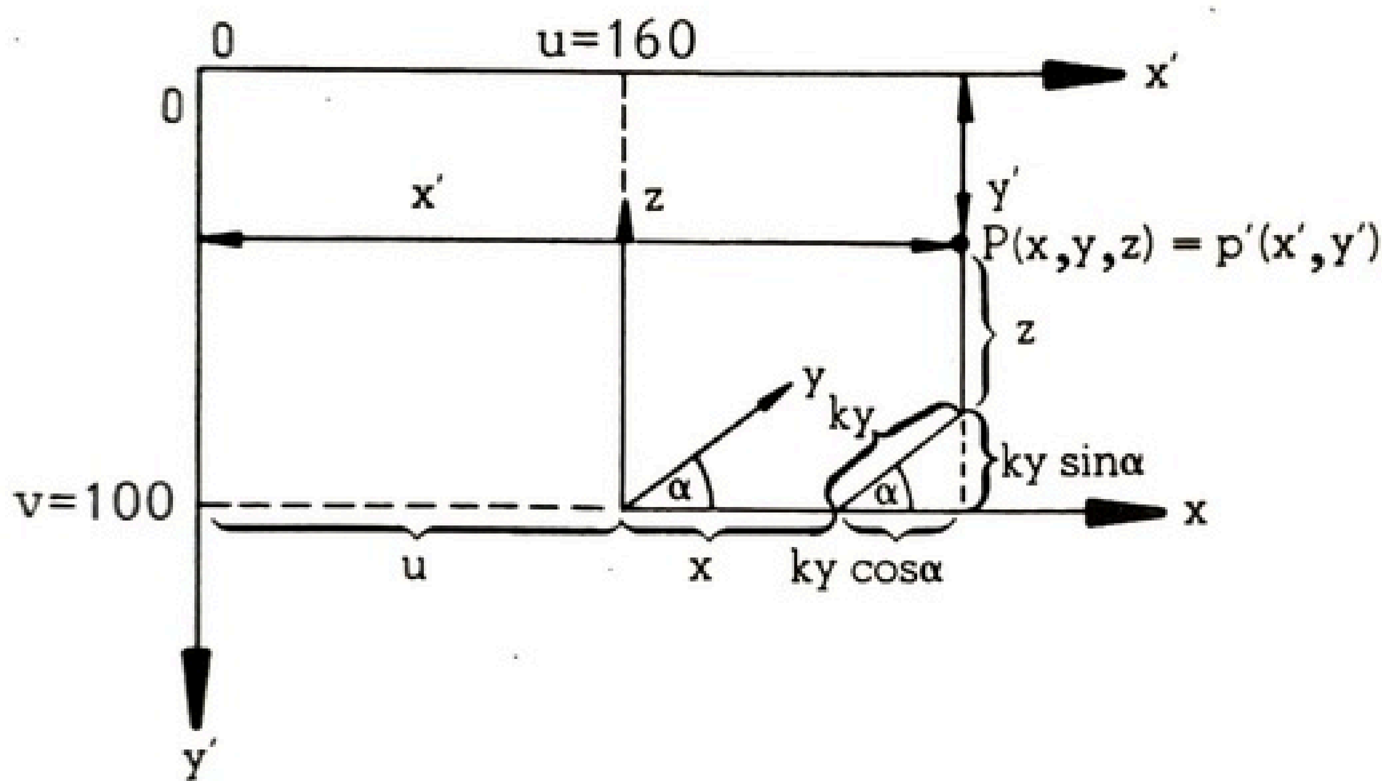
Afleiding van de transformatievergelijkingen

Het projectievlak is ons beeldscherm. We gebruiken ook nu een schermresolutie van 320 bij 200 punten. De oorsprong van het ruimtelijke coördinatensysteem (xyz) moet precies in het midden van het beeldscherm geprojecteerd worden. De oorsprong van het beeldschermcoördinatenstelsel ligt weer in de linkerbovenhoek. De x-as wijst naar rechts; de y-as naar beneden.

De afbeelding van het ruimtelijke punt P(x,y,z) op het punt P'(x',y') in het platte vlak komt tot stand met behulp van de volgende transformatievergelijkingen:

$$x' = U + x + k.y.\cos\alpha \qquad \text{èn}$$
$$y' = V - (k.y.\sin\alpha + z)$$

Hoe we aan deze vergelijkingen komen is uit onderstaande figuur direct (nou ja, direct.....) duidelijk.



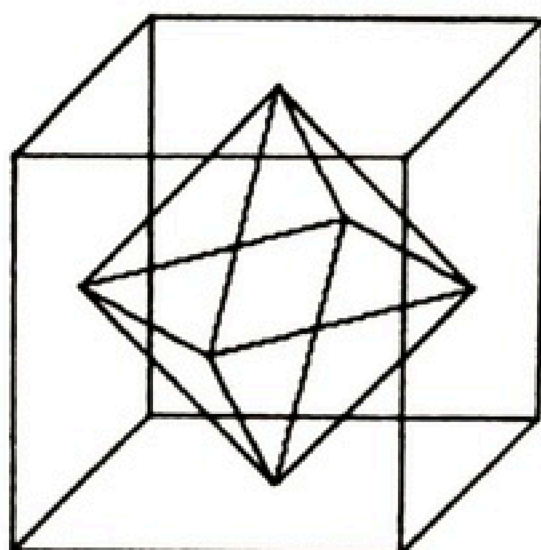
Nemen we C = K*COS(A), S = K*SIN(A) en H = 0.5, dan zijn de transformatieformules in BASIC:

```
X2 = INT(U+X+C*Y+H)
Y2 = INT(V-S*Y-Z+H)
```

Meer wiskunde hebben we voor dit hoofdstuk niet nodig.

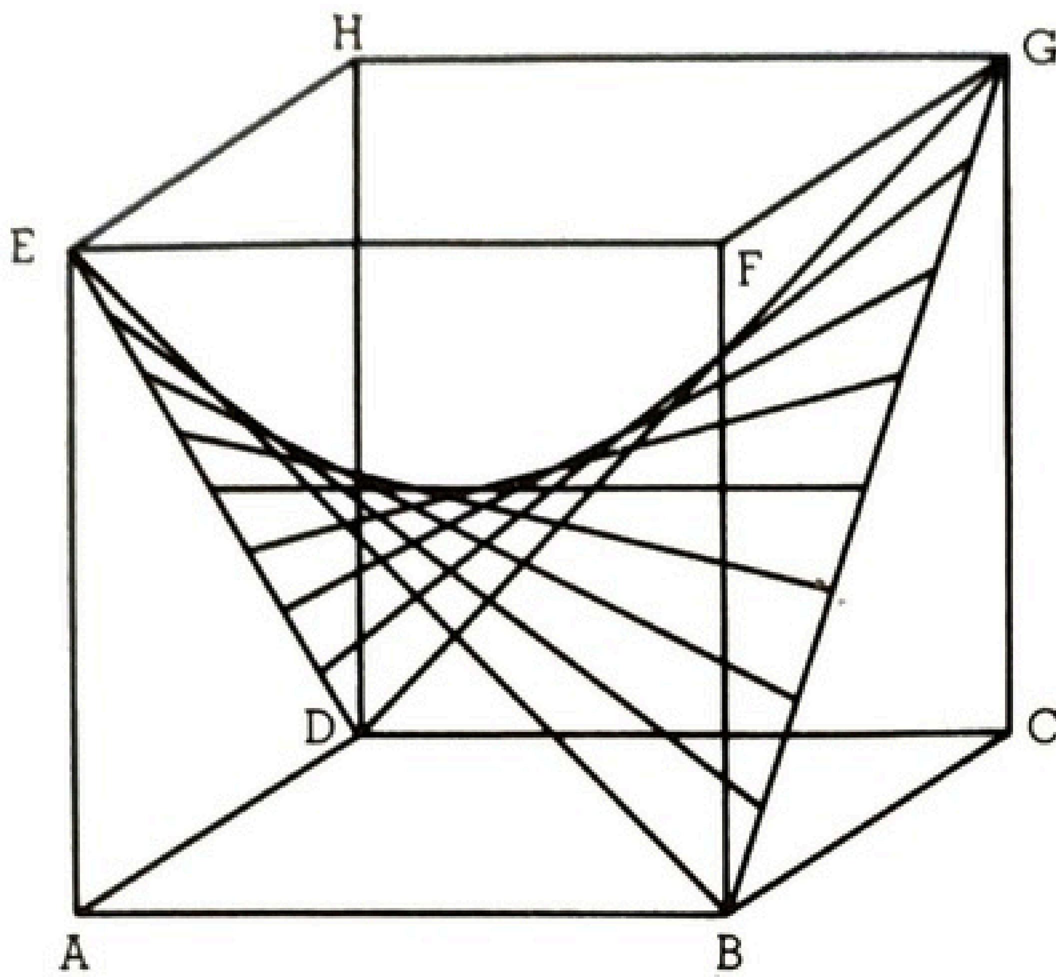
Programma 22 laat zien hoe je van elk, door rechte lijnen begrensd lichaam (polyeder) een projectie kan maken. Hiervoor is het noodzakelijk dat de waarden van de coördinaten x,y,z in alle hoekpunten bekend zijn. Als voorbeeld van het gebruik van dit programma tekenen we een kubus met een ingeschreven achtvlak (oktaëder). De gevolgde programmeertechniek leidt ook bij een viervlak, prisma of pyramide tot het gewenste resultaat.

We nemen aan dat het middelpunt van de kubus samenvalt met de oorsprong van het ruimtelijke coördinatensysteem. Om het programma 'sneller' te maken nemen we de coördinaten van de acht hoekpunten ABCDEFGH van de kubus en de coördinaten van de zes hoekpunten IJKLMN van de achthoek op in DATAregels. Het tekenen van de ribben wordt door de letterstring Z$ bestuurd. Zo betekent Z$ = "ABBCCDDA" dat de computer van A naar B, van B naar C, van C naar D en van D naar A rechte lijnen moet trekken. Met de MID$-functie halen we steeds één letter uit de string Z$ en maken er vervolgens met de ASC-functie een getal van (A=1, B=2, C=3, ...). Hiermee moet de werking van het programma duidelijk zijn. Kies voor alpha 45° en voor k 0.5; dat geeft de mooiste projectie.

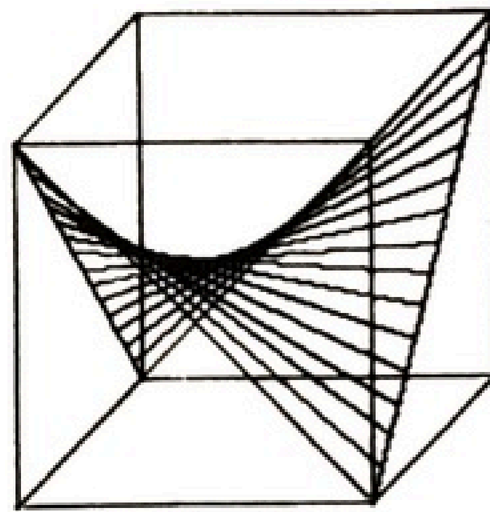


```
100 REM PROGRAMMA 22 KUBUS MET 8-VLAK
110 PRINT CHR$(147)
120 INPUT "ALPHA IN GRADEN (45)    "; A
130 INPUT "VERKLEININGSFACTOR (,5)"; K
140 U=160:V=100:H=0.5:RD=π/180
150 W=A*RD : C=K*COS(W) : S=K*SIN(W)
160 DIM X(14),Y(14),Z$(2)
170 FOR J=1 TO 14
180 :  READ X,Y,Z
190 :  X(J)=INT(U+X+C*Y+H)
200 :  Y(J)=INT(V-S*Y-Z+H)
210 NEXT J
220 PRINT CHR$(147)
230 HIRES 0,1
240 FOR N=1 TO 2
250 :  READ Z$(N) : L=LEN(Z$(N))
260 :  FOR M=1 TO L-1 STEP 2
270 :     I=ASC(MID$(Z$(N),M,1))-64
280 :     J=ASC(MID$(Z$(N),M+1,1))-64
290 :     LINE X(I),Y(I),X(J),Y(J),1
300 :  NEXT M
310 NEXT N
320 GET A$ : IF A$="" THEN 320
330 END
340 :  DATA -60,-60,-60,60,-60,-60
350 :  DATA  60, 60,-60,-60,60,-60
360 :  DATA -60,-60, 60,60,-60, 60
370 :  DATA  60, 60, 60,-60, 60,60
380 :  DATA   0,0,-60,0,-60,  0,60,0,0
390 :  DATA   0, 60,  0,-60,0,0,0,0,60
395 :
400 :  DATA "ABBCCDDAAEBFCGDHEFFGGHHE"
410 :  DATA "IJIKILIMJKKLLMMJJNKNLNMN"
```

In programma 23 wordt in een kubus een zadelvlak getekend. Het geeft het idee van een gekromd vlak in een kubus.



Bekijk de bovenstaande figuur. De diagonalen BG en ED zijn elk in acht (n) gelijke stukken verdeeld en de zo ontstane punten zijn paarsgewijs door een rechte lijn met elkaar verbonden. Kies in het programma voor n de waarde 18.



```
100 REM PROGRAMMA 23 KUBUS MET ZADELVLAK
110 PRINT CHR$(147)
120 INPUT "ALPHA IN GRADEN (45)    "; A
130 INPUT "VERKLEININGSFACTOR (,5)"; K
140 INPUT "HOEVEEL LIJNEN (32)     "; N
150 U=160:V=100:H=0.5:RD=π/180
160 W=A*RD : C=K*COS(W) : S=K*SIN(W)
170 DIM X(8),Y(8)
180 FOR J=1 TO 8
190 :  READ X,Y,Z
200 :  X(J)=INT(U+X+C*Y+H)
210 :  Y(J)=INT(V-S*Y-Z+H)
220 NEXT J
230 PRINT CHR$(147)
240 HIRES 0,1
250 READ Z$ : L=LEN(Z$)
260 FOR M=1 TO L-1 STEP 2
270 :  I=ASC(MID$(Z$,M,1))-64
280 :  J=ASC(MID$(Z$,M+1,1))-64
290 :  LINE X(I),Y(I),X(J),Y(J),1
300 NEXT M
310 FOR J=0 TO N
320 :  X1=INT(X(2)+J*(X(7)-X(2))/N+H)
330 :  Y1=INT(Y(2)+J*(Y(7)-Y(2))/N+H)
340 :  X2=INT(X(5)+J*(X(4)-X(5))/N+H)
350 :  Y2=INT(Y(5)+J*(Y(4)-Y(5))/N+H)
360 :  LINE X1,Y1,X2,Y2,1
370 NEXT J
380 LINE X(2),Y(2),X(7),Y(7),1
390 LINE X(5),Y(5),X(4),Y(4),1
400 GET A$ : IF A$="" THEN 400
410 END
420 :  DATA -60,-60,-60,60,-60,-60
430 :  DATA  60, 60,-60,-60,60,-60
440 :  DATA -60,-60, 60,60,-60, 60
450 :  DATA  60, 60, 60,-60, 60,60
460 :  DATA "ABBCCDDAAEBFCGDHEFFGGHHE"
```

Met programma 24 kunnen we, naar keuze, cilinders, kegels of afgeknotte kegels tekenen. Als we voor R1 en R2 dezelfde waarde invoeren, krijgen we een cilinder. Als R2 kleiner is dan R1 krijgen we een afgeknotte kegel en voor R2 = 0 krijgen we een kegel.

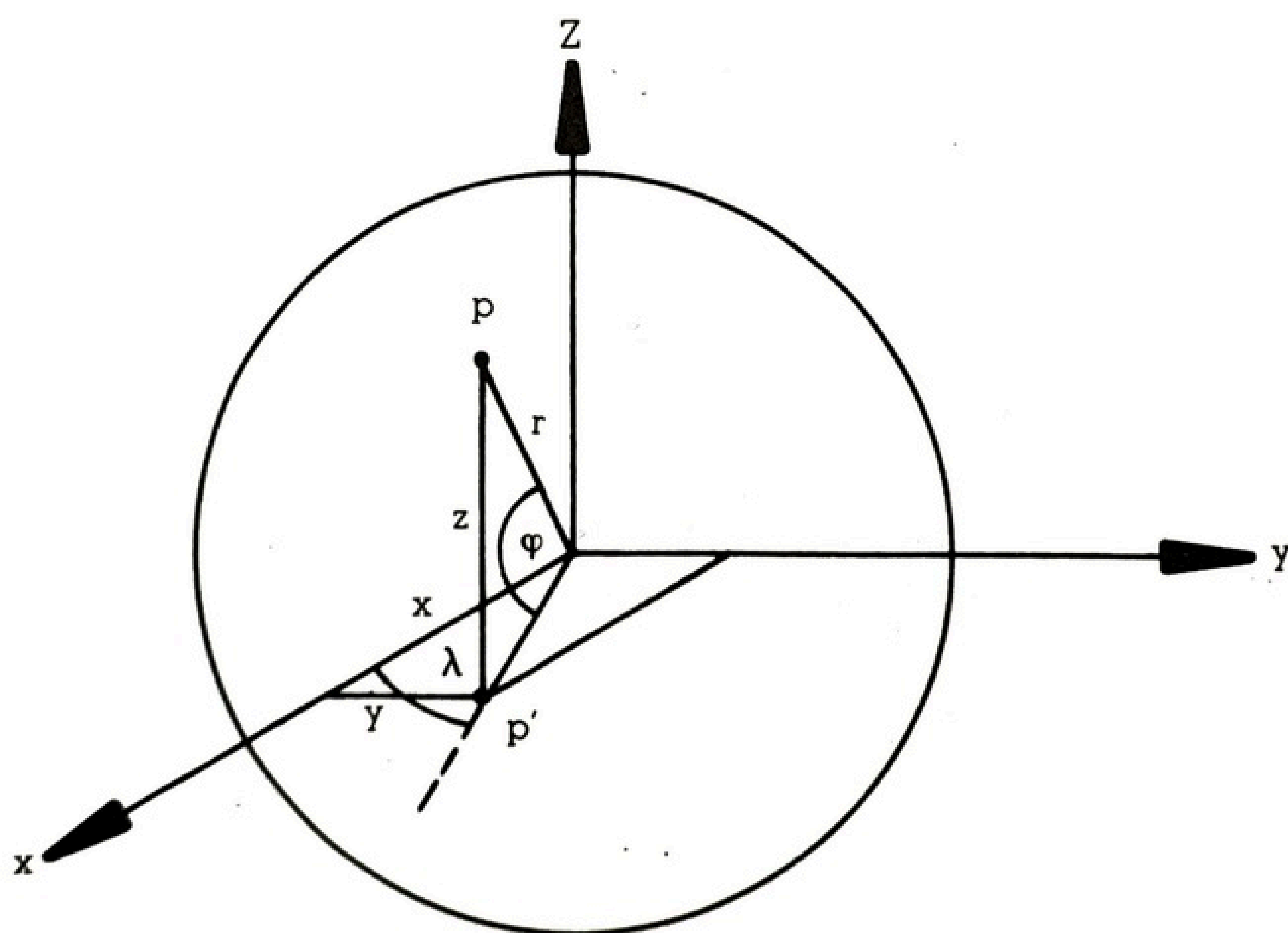
We plaatsen het lichaam zo dat het grondvlak bij z = -60 ligt en het bovenvlak (de top) bij z = +60. Het programma tekent zeven breedtecirkels met een onderlinge afstand van z = 20 en 16 lijnen op de kegel- (of cilinder-) mantel loodrecht op de breedtecirkels. Tot slot wordt verticaal de z-as getekend.

```
100 REM PROGRAMMA 24 CYLINDER,KEGELS
110 PRINT CHR$(147)
120 INPUT "ALPHA IN GRADEN (45)    "; A
130 INPUT "VERKLEININGSFACTOR (.5)"; K
140 INPUT "STRALEN R1 EN R2      "; R1,R2
150 U=160:V=100:H=0.5:RD=π/180:DR=(R1-R2)/6
160 W=A*RD :C=K*COS(W) :S=K*SIN(W) :N=0
170 PRINT CHR$(147)
180 HIRES 0,1
190 FOR Z=-60 TO 60 STEP 20
200 :   R=R1-N*DR
210 :   FOR W=0 TO 360 STEP 3
220 :      W1=W*RD
230 :      X=R*COS(W1) : Y=R*SIN(W1)
240 :      IF W1<>0 THEN 270
250 :      X1=INT(U+X+C*Y+H):Y1=INT(V-S*Y-Z+H)
260 :      GOTO 310
270 :      X2=INT(U+X+C*Y+H)
280 :      Y2=INT(V-S*Y-Z+H)
290 :      LINE X1,Y1,X2,Y2,1
300 :      X1=X2 : Y1=Y2
310 :   NEXT W
320 :   N=N+1
330 NEXT Z
340 FOR W=0 TO 360 STEP 23
350 :   W1=W*RD
360 :   X=R1*COS(W1) : Y=R1*SIN(W1)
370 :   X1=INT(U+X+C*Y+H):Y1=INT(V-S*Y+60+H)
380 :   X=R2*COS(W1) : Y=R2*SIN(W1)
390 :   X2=INT(U+X+C*Y+H):Y2=INT(V-S*Y-60+H)
400 :   LINE X1,Y1,X2,Y2,1
410 NEXT W
420 LINE U,0,U,200,1
430 GET A$ : IF A$="" THEN 430
440 END
```



Erg leuk is het tekenen van een bol met breedtelijnen en meridianen. Zoals bekend is kunnen we elk punt op het boloppervlak eenduidig vastleggen met de straal van de bol (r), de geografische breedte (φ) en de geografische lengte (λ) (zie figuur). Het omzetten van deze bolcoördinaten r,φ,λ in cartesische coördinaten (x,y,z) geschiedt met de volgende drie vergelijkingen:

$x = r\cos\varphi\cos\lambda$
$y = r\cos\varphi\sin\lambda$
$z = r\sin\varphi$



De bolvergelijking in cartesische vorm is overigens $x^2+y^2+z^2=r^2$.

In programma 25 wordt de hoek φ door de W en de hoek λ door P vertegenwoordigd. Kies voor alpha 90° en voor k 0.5. Andere waarden vervormen de bol.

```
100 REM PROGRAMMA 25 BOL
110 PRINT CHR$(147)
120 INPUT "ALPHA IN GRADEN (45)     "; A
130 INPUT "VERKLEININGSFACTOR (.5)"; K
140 INPUT "STRAAL, MAXIMAAL 80      "; R
150 U=160:V=100:H=0.5:RD=π/180
160 W=A*RD :C=K*COS(W) :S=K*SIN(W)
170 PRINT CHR$(147)
180 HIRES 0,1
190 FOR W=-90 TO 90 STEP 15
200 :   W1=W*RD : R1=R*COS(W1)
210 :   FOR P=0 TO 360 STEP 3
220 :      P1=P*RD:X=1.15*R1*COS(P1)
230 :      Y=R1*SIN(P1):Z=R*SIN(W1)
240 :      IF P=0 THEN X1=INT(U+X+H):
           Y1=INT(V-Z+H) : GOTO 290
250 :      X2=INT(U+X+C*Y+H)
260 :      Y2=INT(V-S*Y-Z+H)
270 :      LINE X1,Y1,X2,Y2,1
280 :      X1=X2 : Y1=Y2
290 :   NEXT P
300 NEXT W
310 FOR P=0 TO 180 STEP 15
320 :   P1=P*RD
330 :   FOR W=0 TO 360 STEP 3
340 :      W1=W*RD : R1=R*COS(W1)
350 :      X=1.15*R1*COS(P1)
360 :      Y=R1*SIN(P1):Z=R*SIN(W1)
370 :      IF W=0 THEN X1=INT(U+X+C*Y+H):
           Y1=INT(V-S*Y-Z+H):GOTO 420
380 :      X2=INT(U+X+C*Y+H)
390 :      Y2=INT(V-S*Y-Z+H)
400 :      LINE X1,Y1,X2,Y2,1
410 :      X1=X2 : Y1=Y2
420 :   NEXT W
430 NEXT P
440 GET A$ : IF A$="" THEN 440
450 END
```



Een geliefd demonstratieprogramma van computerleveranciers is een om zijn as draaiend driedimensionaal lichaam. Meestal wordt als 'draai-as' de z-as gekozen, die dan samenvalt met de lengte-as van het lichaam.

Microcomputers die alleen een BASIC-vertolker bezitten zijn te langzaam om zo'n draaiing real-time uit te voeren. Er zijn namelijk nogal ingewikkelde trigonometrische berekeningen voor nodig, die tamelijk veel tijd in beslag nemen. Daarnaast duurt het tekenen van het lichaam in een bepaalde stand in BASIC zo'n 1 à 2 seconden. Zouden we in een BASIC-programma steeds het lichaam tekenen, uitwissen, draaien, tekenen, uitwissen, draaien, ... enz., dan zien we het draaiende lichaam in plaats van een vloeiende beweging een schokkerige beweging maken. De beste oplossing voor dit probleem is de volgende.

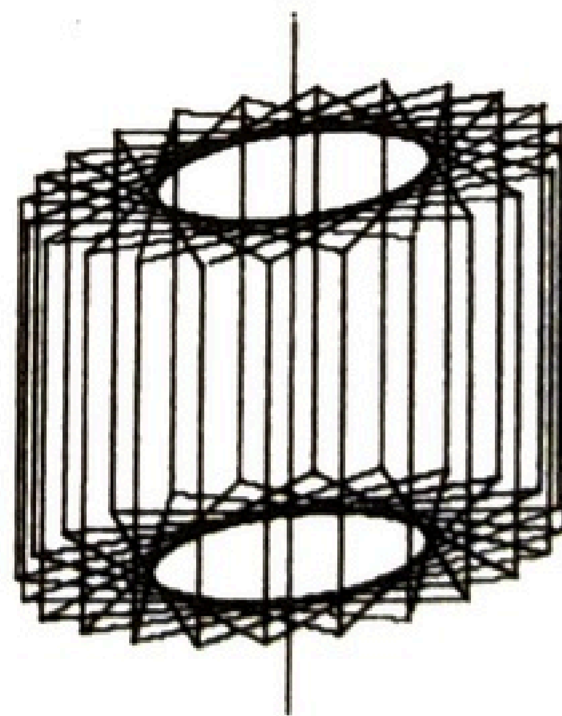
Deel de totale middelpuntshoek van 360° in n even grote hoeken. Voor elk van deze hoeken

$$\omega_1 = \frac{360°}{n}, \quad \omega_2 = 2.\omega_1, \; \ldots\ldots, \; \omega_n = n.\omega_1 = 360°$$

berekenen we van te voren de coördinaten van de hoekpunten van het lichaam. Voor elke stand slaan we deze hoekpuntscoördinaten in een array op. Al deze berekeningen voeren we in BASIC uit bij een leeg beeldscherm. Als de berekeningen klaar zijn wordt een machinetaalprogramma uitgevoerd dat de eerste groep coördinaten uit de array haalt, het lichaam tekent, na een bepaalde tijd het beeldscherm wist, de volgende groep coördinaten ophaalt, ... enz. Deze oplossing heeft een redelijk vloeiende draaiing tot gevolg. Dit programma is geschreven op een CBM 3016 die een 6502-microprocessor bezit. Hier doen we echter anders (zie volgende pagina).

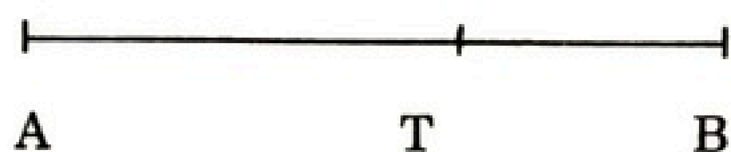
Programma 26 laat een driezijdig prisma om de z-as draaien. De z-as is zowel de lengte-as als de symmetrie-as van het prisma. Het programma tekent een prisma en wacht vervolgens met het draaien en opnieuw tekenen tot we een toets indrukken. Elk volgend prisma wordt over zijn voorgangers heen getekend. We kunnen het draaien dus vertraagd gadeslaan.

```
100 REM PROGRAMMA 26 DRAAIEND PRISMA
110 PRINT CHR$(147)
120 INPUT "ALPHA IN GRADEN (45)    "; A
130 INPUT "VERKLEININGSFACTOR (.5)"; K
140 INPUT "DRAAIHOEK IN GRAGEN(45)"; OM
150 U=160:V=100:H=0.5:RD=π/180:R=80
160 W=A*RD :C=K*COS(W) :S=K*SIN(W)
170 PRINT CHR$(147)
180 HIRES 7,8
190 LINE U,0,U,200,1
200 DIM X(7),Y(7)
210 FOR W=0 TO 360 STEP OM
220 :   FOR J=0 TO 3
230 :       W1=(W+J*120)*RD
240 :       X=R*COS(W1) : Y=R*SIN(W1)
250 :       X(J)=INT(U+X+C*Y+H)
260 :       Y(J)=INT(V-S*Y+60+H)
270 :       X(J+4)=X(J)
280 :       Y(J+4)=INT(V-S*Y-60+H)
290 :   NEXT J
300 :
310 :   FOR J=0 TO 2
320 :      LINE X(J),Y(J),X(J+1),Y(J+1),1
330 :      LINE X(J),Y(J),X(J+4),Y(J+4),1
340 :      LINE X(J+4),Y(J+4),X(J+5),Y(J+5),1
350 :   NEXT J
360 :   GET A$ : IF A$="" THEN 360
370 NEXT W
380 END
```



Programma 27 tekent de projectie van een regelmatig twintigvlak (ikosaëder). Het is bekend dat er slechts vijf 'echt-regelmatige' veelvlakken (polyeders) zijn, namelijk een tetraëder (regelmatig drievlak), een kubus (regelmatig zesvlak), een octaëder (regelmatig achtvlak), een dodekaëder (regelmatig twaalfvlak) en een ikosaëder (regelmatig twintigvlak). Een ikosaëder heeft 12 hoeken, 30 ribben en 20 vlakken. Een vlak is een gelijkzijdige driehoek. Als we programma 22 willen gebruiken om zo'n ikosaëder te tekenen dan hebben we de coördinaten van alle 12 hoekpunten nodig. Als we het ikosaëder in een speciale stand zetten kunnen we de hoekpuntcoördinaten met behulp van de techniek van de 'gulden snede' relatief eenvoudig berekenen. Wie zich voor de hierachterliggende wiskundige theorie interesseert moet er maar eens een meetkundeboek op naslaan. De 'gulden-snede'-techniek deelt een lijnstuk AB in twee stukken AT en TB op zo'n manier dat de volgende evenredigheid geldt:

AB : AT = AT : TB



A T B

Kiezen we voor AB de lengte 1, dan kunnen we de lengte van AT berekenen; immers

$$\frac{AB}{AT} = \frac{AT}{TB}, \quad \text{dus} \quad \frac{1}{t} = \frac{t}{1-t}, \quad \text{als } t = AT$$

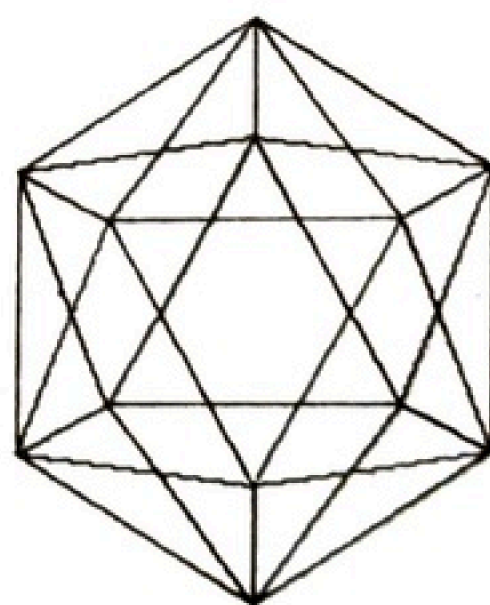
dus dan moet gelden $t^2 = 1-t$ ofwel $t^2+t-1=0$.
Met de alombekende abc-formule berekenen we nu

$$t_{1,2} = \frac{-1 \pm \sqrt{1+4}}{2}$$

De positieve wortel geeft: $t = \frac{-1 + \sqrt{5}}{2}$

Dit is de lengte van het grootste van de twee stukken die we krijgen als we een lijnstuk, ter lengte 1, volgens de 'gulden snede' in twee stukken verdelen. (t = 0,618; $1-t$ = 0,382).
Deze waarde wordt in regel 160 berekend en in de regels 200-310 gebruikt om de hoekpuntcoördinaten van het ikosaëder te berekenen. Om te zorgen dat het twintigvlak voldoende groot getekend wordt zijn alle coördinaten met 80 (F) vermenigvuldigd. Kies voor alpha (A) 90° en voor k (K) de waarde 0.4. Andere waarden vertekenen het beeld. Alpha = 180° en K=0,6 geeft trouwens iets onverwachts!

```
100 REM PROGRAMMA 27 IKOSAEDER
110 PRINT CHR$(147)
120 INPUT "ALPHA IN GRADEN (90)    "; A
130 INPUT "VERKLEININGSFACTOR (.4)"; K
140 U=160:V=100:H=0.5:RD=π/180
150 W=A*RD :C=K*COS(W) :S=K*SIN(W)
160 T=(SQR(5)-1)/2 : F=80
170 PRINT CHR$(147)
180 HIRES 7,8
190 DIM X(12),Y(12),Z(12),Z$(3)
200 X(1)=   0: Y(1)=  F*T:  Z(1)=  -F
210 X(2)=   0: Y(2)= -F*T:  Z(2)=  -F
220 X(3)=   F: Y(3)=    0:  Z(3)=-F*T
230 X(4)=  -F: Y(4)=    0:  Z(4)=-F*T
240 X(5)= F*T: Y(5)=    F:  Z(5)=   0
250 X(6)=-F*T: Y(6)=    F:  Z(6)=   0
260 X(7)= F*T: Y(7)=   -F:  Z(7)=   0
270 X(8)=-F*T: Y(8)=   -F:  Z(8)=   0
280 X(9)=   F: Y(9)=    0:  Z(9)= F*T
290 X(10)= -F: Y(10)=   0: Z(10)= F*T
300 X(11)=  0: Y(11)= F*T: Z(11)=   F
310 X(12)=  0: Y(12)=-F*T: Z(12)=   F
320 FOR N=1 TO 3
330 :  READ Z$(N) : L=LEN(Z$(N))
340 :  FOR M=1 TO L-1 STEP 2
350 :     I=ASC(MID$(Z$(N),M,1))-64
360 :     J=ASC(MID$(Z$(N),M+1,1))-64
370 :     X1=INT(U+X(I)+C*Y(I)+H)
380 :     Y1=INT(V-S*Y(I)-Z(I)+H)
390 :     X2=INT(U+X(J)+C*Y(J)+H)
400 :     Y2=INT(V-S*Y(J)-Z(J)+H)
410 :     LINE X1,Y1,X2,Y2,1
420 :  NEXT M
430 NEXT N
440 GET A$ : IF A$="" THEN 440
450 END
460 : DATA "BCCEEFFDDBABACAEAFAD"
470 : DATA "BHHDDJJFFKKEEIICCGGB"
480 : DATA "GHHJJKKIIGLGLHLJLKLI"
```



5 *Het tekenen van vlakken in de ruimte*

In het vorige hoofdstuk hebben we ons uitvoerig beziggehouden met het tekenen van driedimensionale lichamen. We herhalen hiervan nog eens de belangrijkste punten:

1. We gebruiken de parallelprojectie om een driedimensionaal lichaam met n hoekpunten P(x,y,z) in een plat vlak te tekenen. Voor de projectiehoek alpha (α) gebruiken we bij voorkeur de waarde 45° of 60°. De schuin-naar-achteren lopende ribben worden korter getekend dan ze in werkelijkheid zijn. Als verkleiningsfactor kiezen we vaak 1/2 of 1/3.

2. Elk punt P(x,y,z) van het ruimtelijke lichaam wordt op een punt P'(x',y') in het projectievlak geprojecteerd. De hierbij gebruikte projectieformules zijn:

   $$x' = U + k.y.\cos\alpha$$
   $$y' = V - k.y.\sin\alpha - z$$

   Zoals gebruikelijk zijn U en V de coördinaten van het midden van het beeldscherm. Tot nu toe hebben we U = 160 en V = 100 verondersteld.

In de volgende BASIC-programma's zien we de bovenstaande transformatievergelijkingen in de vorm:

```
XG = INT(U+XX+C*YY+H)
YG = INT(V-S*YY-Z+H)
```

De betekenis van de gebruikte variabelen is:

XG,YG	:	beeldschermcoördinaten x',y'
XX,YY	:	de lopende coördinaten x,y
C	:	K*COS(W*RD)
S	:	K*SIN(W*RD)
W	:	projectiehoek in graden

RD : factor $\pi/180$ voor omrekenen van graden in radialen
K : verkleiningsfactor

Tekenen van driedimensionale functies

Als paradepaardje van menige demonstratie van Hoge-Resolutie-Graphics wordt vaak een programma gebruikt dat een of andere fraaie grafiek van een driedimensionale functie tekent. Als leek vraag je je af hoe ze weten welke functies ze moeten nemen, hoe het tekenen van de grafiek van zo'n functie geprogrammeerd wordt, en hoe ze het programmatechnisch voor elkaar krijgen dat de 'onzichtbare lijnen' ook inderdaad niet getekend worden. Bij dergelijke demonstraties wordt doorgaans geen programmalisting getoond. Mocht dit wel het geval zijn dan is het programma vaak zo machine-afhankelijk dat het omzetten van het programma naar een andere machine lastig, zo niet ondoenlijk is.

In dit hoofdstuk hopen we u antwoord te kunnen geven op de volgende vragen:

1. Hoe vind ik 'mooie' driedimensionale functies?

2. Hoe ziet een algemeen programma voor het tekenen van een dergelijke functie eruit?

3. Hoe onderdruk je het tekenen van de onzichtbare lijnen (hidden lines)?

We beginnen met de definitie van een driedimensionale functie.

Elke functie van de vorm z = f(x,y) heet een driedimensionale functie en vormt een, meestal, gekromd vlak in een driedimensionaal (ruimtelijk) coördinatenstelsel. Speciaal voor niet-wiskundigen volgt nu een voorbeeld van een functie z = f(x,y) en het gekromde vlak dat als 'grafiek' bij de functie hoort.

Stel dat we voor z = f(x,y) de volgende vergelijking nemen:

$$z = e^{-(x^2+y^2)}$$

(e is het grondtal van de natuurlijke logaritme; e = 2,718284183...)

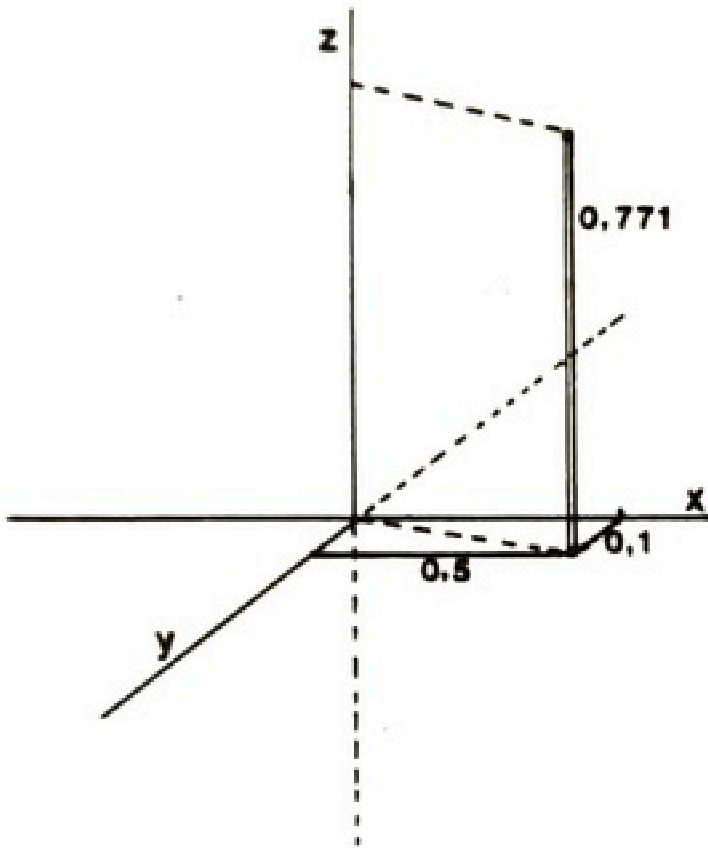
Voor elk punt (x,y) in het (platte) X-Y-vlak kunnen we nu de daarbij horende waarde van z uitrekenen. Als x = 0.5 en y = 0.1 dan is

$z = e^{-(0.25+0.01)}$ ⇒

$z = e^{-0.26}$ ⇒

z = 0.771, want 2,71828... tot de macht -0.26 is 0.771

Zet nu (in gedachten) in het voetpunt P(0,5;0,1) in het X-Y-vlak een staafje met een lengte van 0.771 rechtop. Doe ditzelfde voor nog een groot aantal punten (x,y).



Leg nu over al deze staafjes een elastische folie. Dit (golvende) stuk folie vormt een goed model van het vlak met vergelijking $z = e^{-(x^2+y^2)}$.
Hoe dit eruit zou zien kunt u zien op p.73.

Antwoord op de eerste vraag

Zoek met behulp van programma 7 uit hoofdstuk 2 een willekeurige contine functie die symmetrisch ten opzichte van de y-as is. De grafiek van de functie moet 'bergen en dalen' vertonen (de functie moet maxima en minima hebben). Erg geschikt zijn trigonometrische functies (sin, cos) en combinaties van deze functies. Ook geschikt zijn functies waarin alleen termen voorkomen met 'even-machten' van x en exponentiële functies.

Enkele voorbeelden:

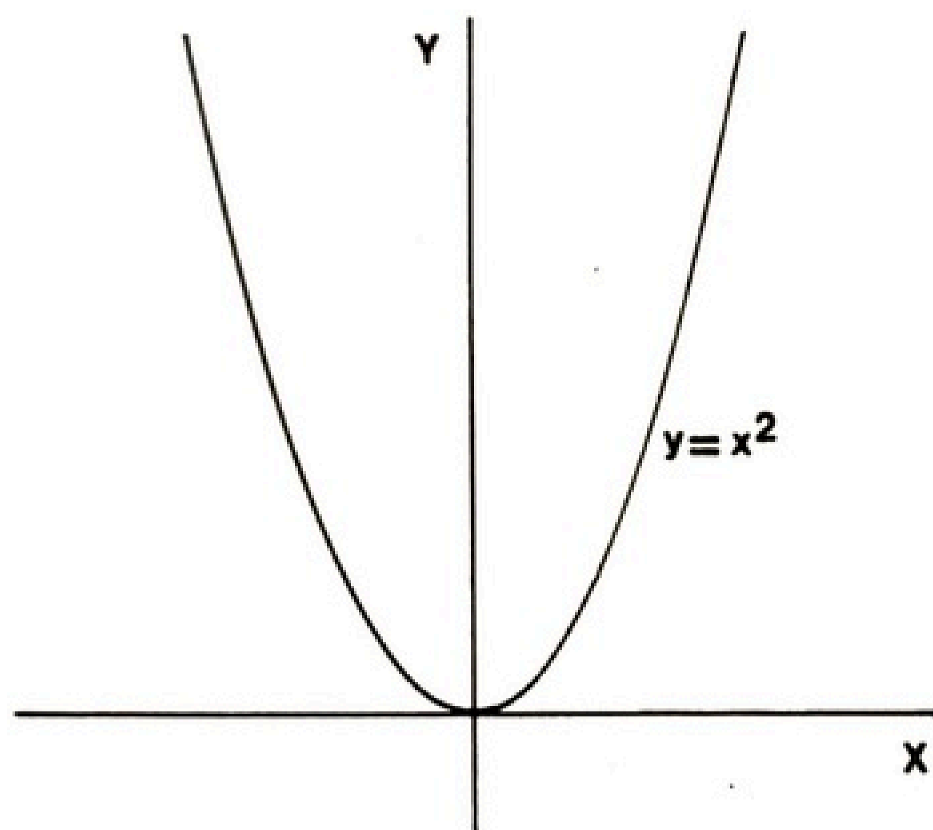
$y = e^{-x^2}$; $y = \sin(x)$

$$y = \cos(x) - \frac{\cos(3x)}{3} + \frac{\cos(5x)}{5} - \frac{\cos(7x)}{7}$$

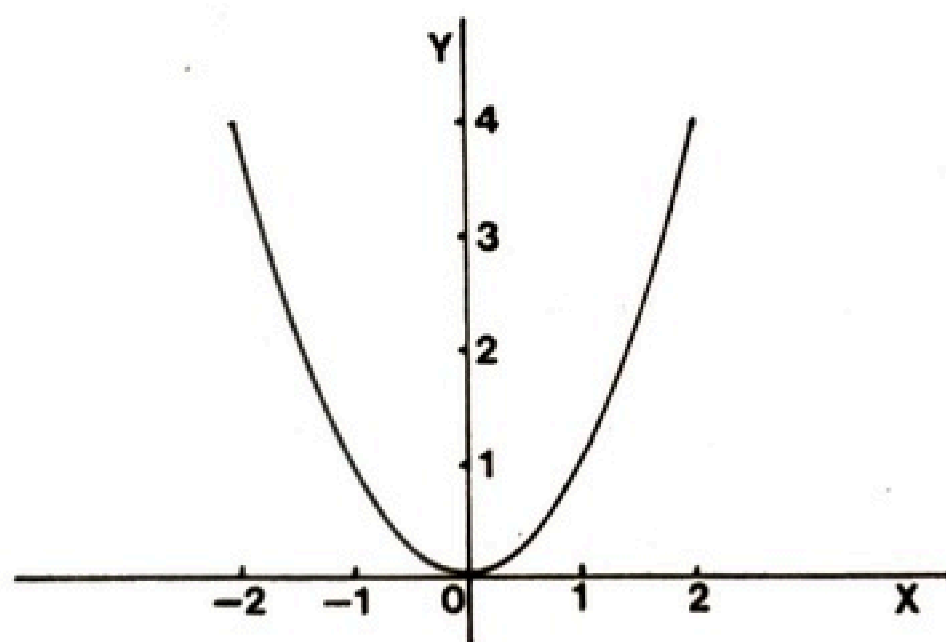
Bekijk nu de grafiek van zo'n functie in het interval $-a \leq x \leq a$. Dit 'stuk grafiek gaan we draaien rond de y-as. Zo ontstaat een, ten opzichte van de y-as, draaisymmetrisch driedimensionaal vlak. Als we nu de y-as als z-as kiezen, dan wordt de vergelijking van een dergelijk draaisymmetrisch ruimtelijk vlak (rond de z-as) van de vorm

$$z = f(r) \qquad \text{met} \quad r = \sqrt{x^2+y^2}$$

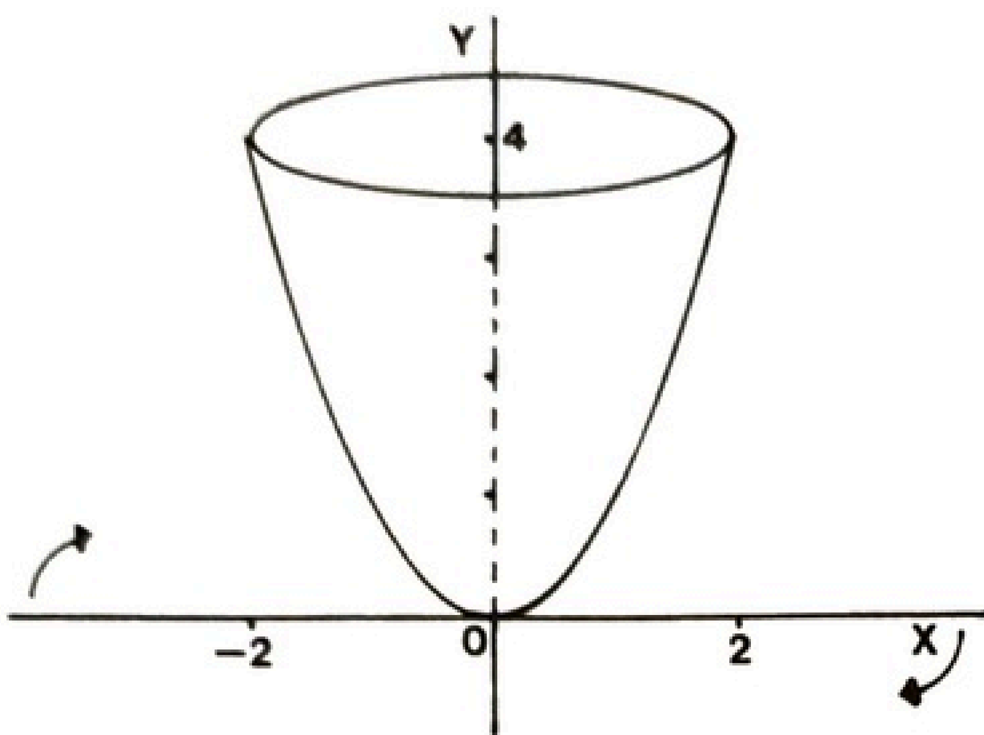
We zullen dit aan een eenvoudig voorbeeld proberen te verklaren. We kiezen als voorbeeld de eenvoudige paraboolvergelijking $y=x^2$.



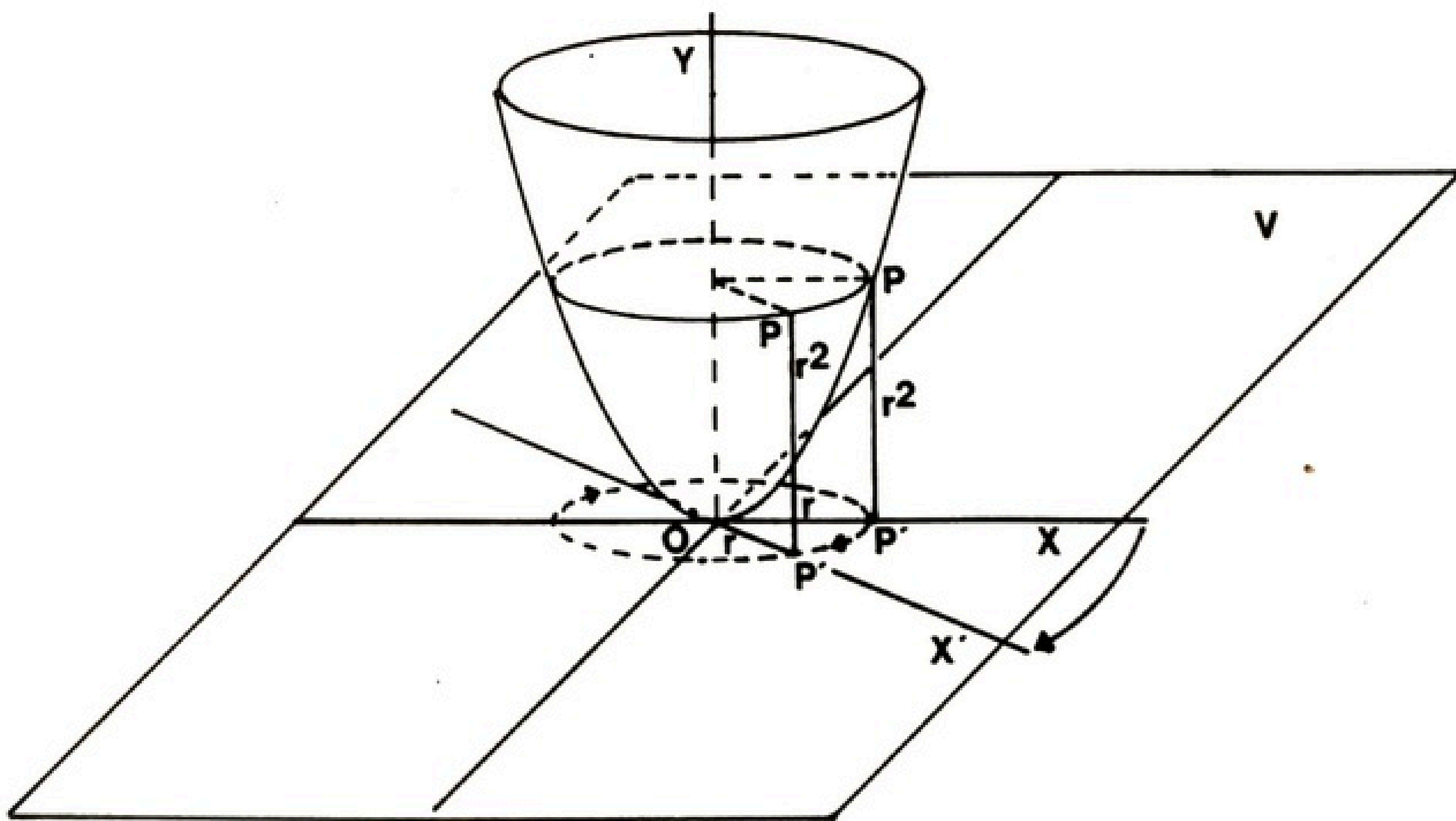
Stel we bekijken de grafiek voor $-2 \leq x \leq 2$:



Als we nu de x-as rond de y-as laten draaien (de x-as komt als het ware loodrecht het papier uit), dan ontstaat een draaisymmetrisch vlak rond de y-as. Dit is een kegel:



Elk punt P op de kegelmantel heeft de eigenschap $y=r^2$, waarbij r de afstand is van het geprojecteerde punt P' tot het punt O. We kunnen het vlak V dat door de ronddraaiende x-as wordt gevormd als volgt tekenen:

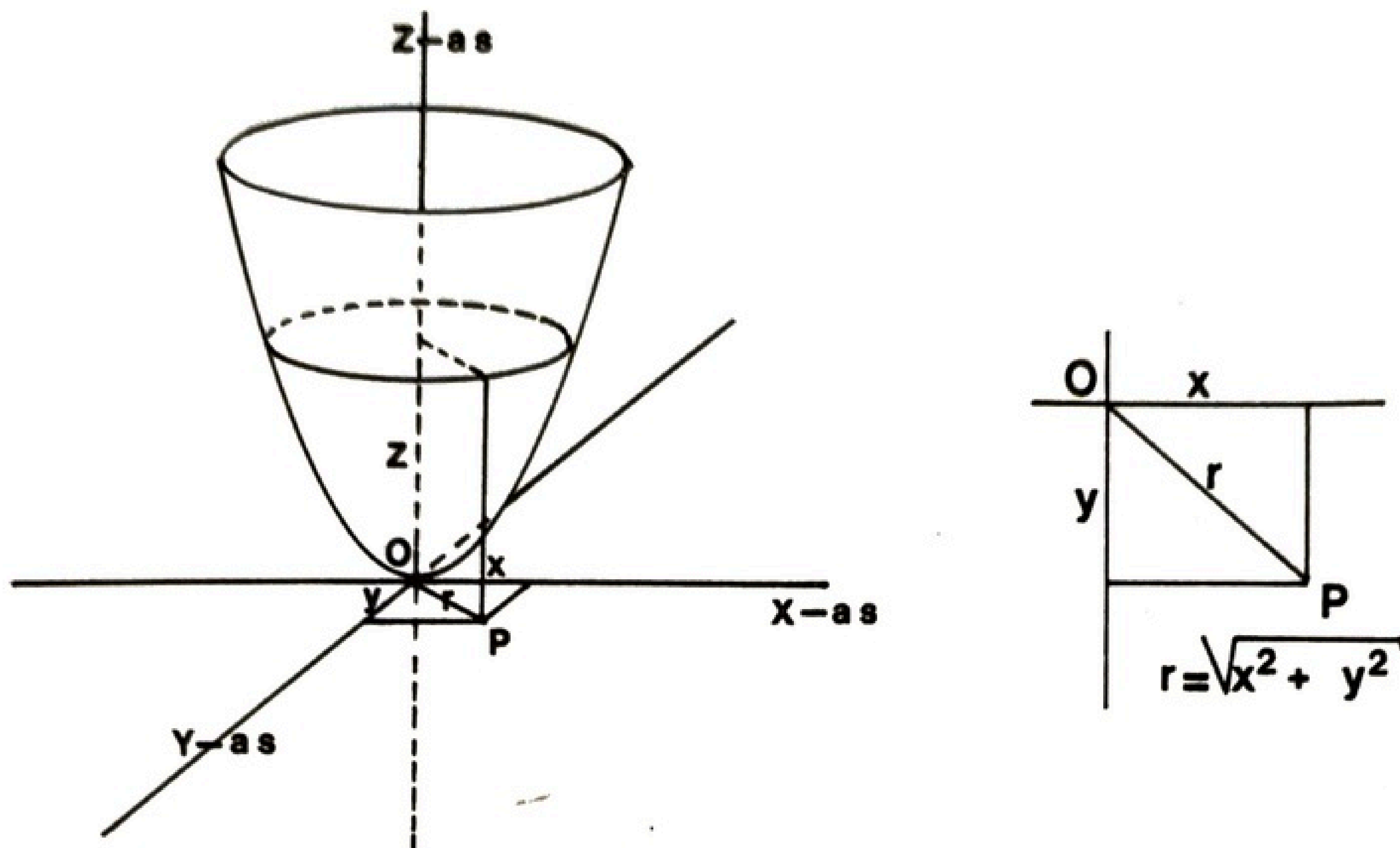


Welnu, als we nu de z-as als y-as nemen en we nemen de y-as in vlak V loodrecht op de x-as, dan zien we dat de afstand r geschreven kan worden als

$r = \sqrt{x^2+y2}$ (stelling van Pythagoras, zie rechts in de volgende tekening)

en nu wordt $y=r^2$ geschreven als $z=r^2$ met $r^2 = x^2+y^2$, dus de vergelijking van de paraboloïde (kegel) wordt dan

$$z = x^2+y^2$$



Dus $y=x^2$ wordt bij draaiing om de y-as en bij een z-as die de plaats van de y-as inneemt $z = x^2+y^2$.
Voor de bovengenoemde functies geldt het volgende:

$y = e^{-x^2}$ wordt $z = e^{-(x^2+y^2)}$

$y = \dfrac{\sin(x)}{x}$ wordt $z = \dfrac{\sin(r)}{r}$

$y = \cos(x) - \dfrac{\cos(3x)}{3} + \dfrac{\cos(5x)}{5} - \dfrac{\cos(7x)}{7}$ wordt

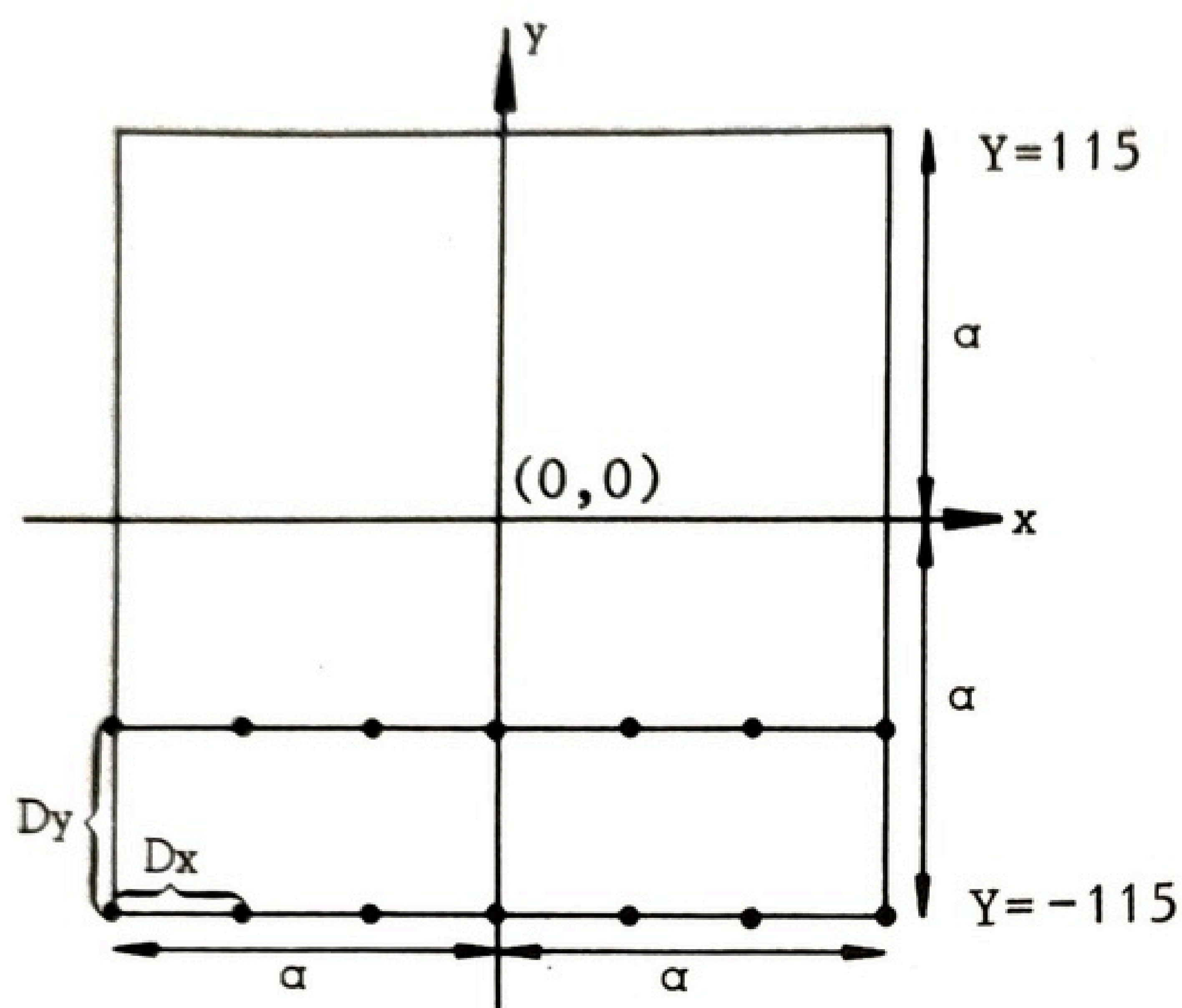
$y = \cos(r) - \dfrac{\cos(3r)}{3} + \dfrac{\cos(5r)}{5} - \dfrac{\cos(7r)}{7}$

met $r = \sqrt{x^2+y^2}$

Voor deze klasse van draaisymmetrische figuren ontwikkelen we een algemeen tekenprogramma.

Antwoord op de tweede vraag

Bekijk de onderstaande tekening. U kijkt recht bovenop het ruimtelijke vlak van een driedimensionale functie. We willen dat het op het grondvlak geprojecteerde vlak een vierkant is met zijden van 2a. Straks zullen we zien dat, als we een mooie tekening van zo'n ruimtelijk vlak op het beeldscherm willen maken, a=115 een heel goede keus is.



Als 'doorgewinterde' programmeur ziet u natuurlijk direct dat het programma voor het tekenen van het vlak een geneste FOR-NEXT (dx,dy) zal bevatten! Als we een vlak als grafiek van z=f(x,y) willen teken, moeten we namelijk voor een aantal combinaties van x en y de functiewaarde z = f(x,y) berekenen. We beginnen met y =-115 en laten x met stapjes dx (bijvoorbeeld dx=3) van -115 naar +115 lopen. Voor elke combinatie x,y met y =-115 berekenen we de functiewaarde z = f(x,y). Zo ontstaan de punten (x,y,z) van het ruimtelijke vlak waarvan de geprojecteerde punten (x,y) in de bovenstaande tekening op de onderste zijde van het vierkant als bolletjes getekend zijn. Nu verhogen we y met dy en laten x weer van -115 tot +115 lopen. Dit geeft de tweede reep van het vlak. Op deze wijze ontstaat het hele vlak, waarvan de punten (x,y,z) natuurlijk nog getransformeerd moeten worden naar beeldschermcoördinaten (x',y').

Een eerste globale opzet voor het tekenen van een 'vierkant'-stuk vlak is:

```
FOR Y=-115 TO 115 STEP DY
   FOR X=-115 TO 115 STEP DX
      GOSUB 1000
      :
   NEXT X
NEXT Y
```

In de subroutine 1000 wordt voor een punt (x,y) de waarde van z = f(x,y) berekend. Met de bekende transformaties wordt vervolgens het punt P(x,y,z) overgebracht naar een punt P'(x',y') op het beeldscherm. Nu kan het berekende punt P(x,y,z), dat in het vlak ligt, als het punt P'(x',y') op het scherm worden getekend.

Jammer genoeg lenen niet alle driedimensionale functies zich voor de intervallen $-115 \leq x \leq 115$ en $-115 \leq y \leq 115$; vandaar dat de waarde voor a met een INPUT-opdracht ingelezen wordt. Als we in het programma toch $-115 \leq XX \leq 115$ en $-115 \leq YY \leq 115$ kiezen, moet de ingetoetste waarde A als volgt gebruikt worden:

$$X = XX.\frac{A}{115} \quad \text{en} \quad Y = YY.\frac{A}{115}$$

dat wil zeggen er geldt: $-A \leq X \leq A$ en $-A \leq Y \leq A$

De z-waarden van met name de trigonometrische driedimensionale functies zal tussen -1 en 1 liggen. Om deze waarden wat 'op te blazen' kan een vermenigvuldigingsfactor (K1) worden ingetoetst. Goede waarden voor K1 liggen tussen 30 en 80.

We kunnen ons programma nu als volgt opschrijven:

```
    DIT IS EEN PROGRAMMA-OPZET VOOR
    HET TEKENEN VAN DE FUNCTIE Z=F(X,Y)

INPUT "ALPHA IN GRADEN (45-135)"; W
INPUT "VERKLEININGSFACTOR (.5-.75)"; K
INPUT "RECHTERGRENS VOOR X(>0)"; A
INPUT "VERGROTINGSFACTOR (30-80)"; K1
U=160:V=100:H=0.5:RD=π/180
C=K*COS(W*RD) : S=K*SIN(W*RD)
DX=3 : DY=5 : AF=A/115
PRINT CHR$(147)
HIRES 7,8
FOR YY=-115 TO 115 STEP DY
   Y=YY*AF
   FOR XX=-115 TO 115 STEP DX
      X=XX*AF : GOSUB 1000
      XG=INT(U+XX+C*YY+H)
      YG=INT(V-S*YY-Z +H)
      REM TEKEN PUNT P(XG,YG)
   NEXT XX
NEXT YY
GET A$ : IF A$="" THEN 430
END
    BESCHRIJVING VAN Z=F(X,Y)
```

Zoals beloofd laten we nu zien waarom de intervallen $-115 \leq XX \leq 115$ en $-115 \leq YY \leq 115$ zo geschikt zijn om driedimensionale functies te tekenen met een hoog oplossend vermogen. We willen graag de fraaie projectie met α=45° en k=0,5 gebruiken. Wat worden dan de beeldschermcoördinaten XG en YG voor de hoekpunten 'linksonder' en 'rechtsboven' van het vierkant op p.69? Deze punten bepalen immers of de hele grafiek op ons beeldscherm van 320 bij 200 puntjes getekend kan worden. Voor K1 nemen we 50, dat ligt zo'n beetje tussen 30 en 80! Nemen we voor de z-waarde ook 50, dan gaat het punt (x,y,z) met coördinaten (-115,-115,50) over in het beeldschermpunt (XG,YG) met

XG = INT(160-115-0,5.115.cos(45)+0,5) ⇒ XG = 4
YG = INT(100+0,5.115.sin(45)-50+0,5) ⇒ YG = 91

Deze waarden liggen inderdaad binnen ons 'graphic-scherm' ($0 \leq XG \leq 320$ en $0 \leq YG \leq 100$) en de 'breedte' (320) van het scherm wordt heel goed benut, kijk maar naar de coördinaten (XG,YG) voor het punt (115,115,50) rechtsboven:

XG = INT(160+115+0,5.115.cos(45)+0,5) ⇒ 316
YG = INT(100-0,5.115.sin(45)-50+0,5) ⇒ 9

We benutten dus haast de hele schermbreedte ($4 \leq XG \leq 316$) voor het tekenen van de driedimensionale figuur. In programma 28 (regel 270) en in programma 29 (regel 305) is rekening gehouden met eventuele negatieve XG-waarden en XG-waarden groter dan 320. Als dit voorkomt, worden deze waarden op respectievelijk 0 en 320 gezet, zodat het programma niet door een foutmelding afbreekt.

Op de onder- en bovenrand (YY=-115 en YY=+115) gelden vaak heel kleine z-waarden, zodat bij α=45°, k=0,5 en z=0 de grafiek op het scherm zal liggen tussen

z-waarde ↓

YG = INT(100+0,5.115.sin(45)-0+0,5) ⇒ 141
en YG = INT(100-0,5.115.sin(45)-0+0,5) ⇒ 59

Ons tekenvlak is dus ongeveer $4 \leq XG \leq 316$ èn $59 \leq YG \leq 141$.

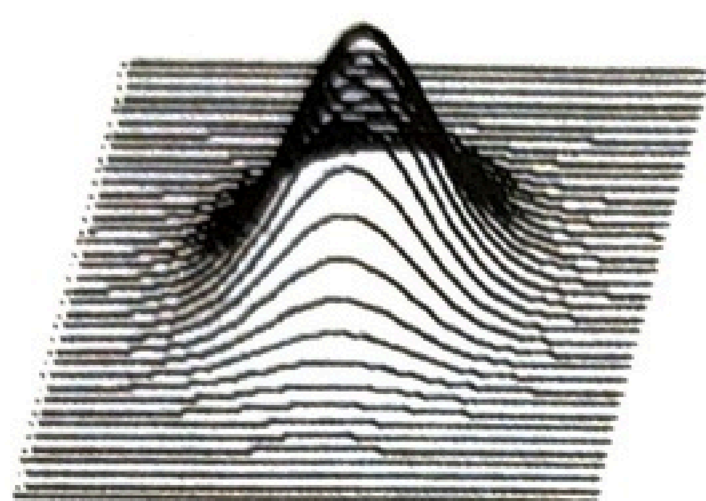
In het bovenstaande programmavoorstel wordt gebruik gemaakt van 'puntgraphics'. Hierbij worden de punten puntje voor puntje getekend. Om een enigszins acceptabele tekening te krijgen moeten erg veel 'puntjes' berekend worden (DX=1,DY=1). Dit duurt in BASIC (geneste lussen en vele trigonometrische berekeningen) erg lang. Voor de onderstaande tekeningen betekent dit al snel een tekentijd van meer dan 30 minuten, soms wel een uur. Dit is weinig bevredi-

gend! Veel sneller gaat het als we de 'vectorgraphics'-methode gebruiken. Deze techniek hebben we in alle voorgaande programma's gebruikt; we verbinden steeds twee naast elkaar liggende punten door een recht lijntje. Hierbij hoeven we lang niet zoveel punten te berekenen als bij de 'puntgraphics'-techniek. Bovendien ziet de tekening er beter uit.

Als antwoord op de tweede vraag komen we dan met het volgende algemene programma voor het tekenen van draaisymmetrische ruimtelijke vlakken.

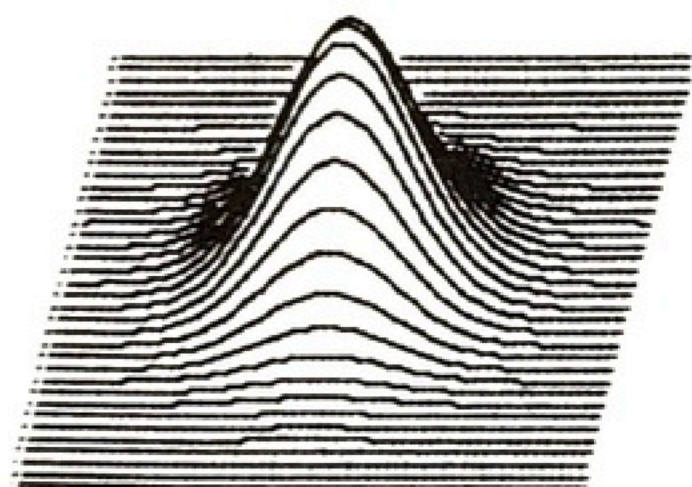
```
100 REM PROGRAMMA 28 GRAFIEK VAN Z=F(X,Y)
110 PRINT CHR$(147)
120 INPUT "ALPHA IN GRADEN (45-135)"; W
130 INPUT "VERKLEININGSFACTOR (.5-.75)"; K
140 INPUT "RECHTERGRENS VOOR X(>0)"; A
150 INPUT "VERGROTINGSFACTOR (30-80)"; K1
160 U=160:V=100:H=0.5:RD=π/180
170 C=K*COS(W*RD) : S=K*SIN(W*RD)
180 DX=3 : DY=5 : AF=A/115
190 PRINT CHR$(147)
200 HIRES 7,8
210 FOR YY=-115 TO 115 STEP DY
220 :   Y=YY*AF
230 :   FOR XX=-115 TO 115 STEP DX
240 :      X=XX*AF : GOSUB 1000
250 :      XG=INT(U+XX+C*YY+H)
260 :      YG=INT(V-S*YY-Z +H)
270 :      IF XG<0 THEN XG=0:IF XG>320 THEN XG=320
280 :      IF YG<0 OR YG>200 THEN PRINT "FOUTE K1":STOP
290 :      IF XX=-115 THEN X1=XG:Y1=YG:GOTO 330
300 :      X2=XG : Y2=YG
310 :      LINE X1,Y1,X2,Y2,1
320 :      X1=X2 : Y1=Y2
330 :   NEXT XX
340 NEXT YY
350 GET A$ : IF A$="" THEN 350
360 END
1000 Z=K1*EXP(-X*X-Y*Y)
1100 RETURN
```

Met dit programma maakt u de tekening op p.73. Voor de plotter die wij gebruikt hebben golden hierbij de volgende waarden: W=75°, K=0.75, A=3 en K1=75. U kunt op uw Commodore ook deze combinatie gebruiken. Veel mooier is echter de combinatie W=45°, K=0.5, A=3 en K1=80. Dit kan op de Commodore 64 (zie p.71).



$z = e^{-(x^2+y^2)}$

Omdat onze plotter horizontaal slechts 220 puntjes kan weergeven, kunnen we de tekeningen niet in het fraaie perspectief $\alpha=45°$ laten zien. Op het beeldscherm van uw computer kan dat wel. Toch zult u over de bovenstaande tekening niet tevreden zijn. De voorgrond is goed, maar in het achterste stuk zijn ook alle lijnen die voor ons onzichtbaar zijn getrokken. In onderstaande tekening zien we het resultaat met dezelfde waarden voor W, K, A en K1, maar waarin de niet-zichtbare lijnen ook niet getekend zijn. Hoe krijgen we dit voor elkaar?



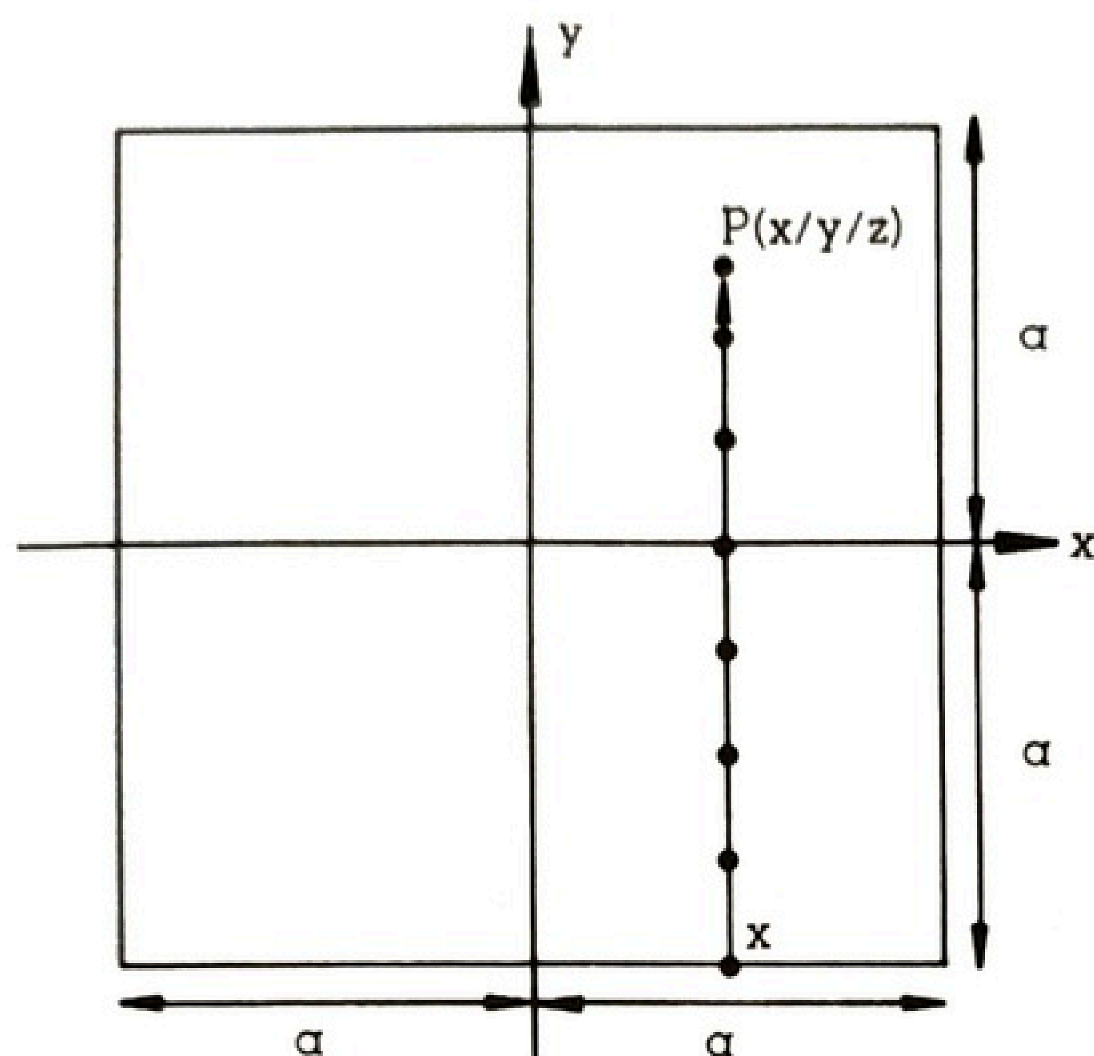
$z = e^{-(x^2+y^2)}$

Antwoord op de derde vraag

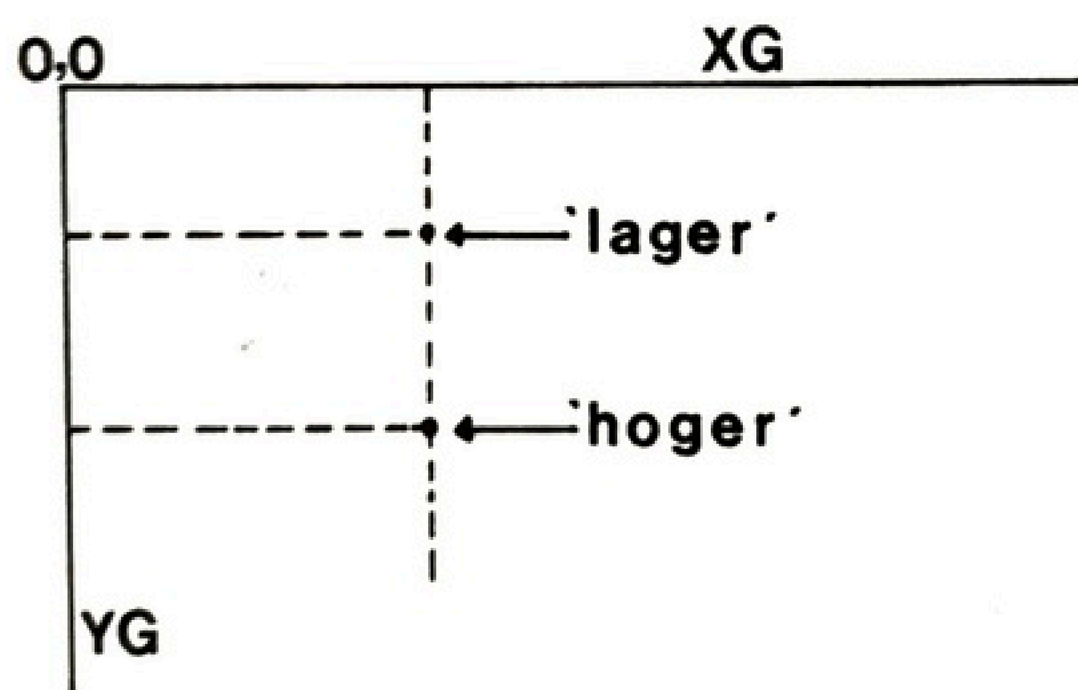
Bekijk de onderstaande tekening. Een punt P(x,y,z) van het te tekenen vlak is alleen dan zichtbaar wanneer:

a. het punt 'hoger' ligt dan alle voorgaande punten met dezelfde x-coördinaat of
b. het punt 'lager' ligt dan alle voorgaande punten met dezelfde x-coördinaat.

Vanzelfsprekend geldt dit ook voor de op het beeldscherm geprojecteerde punten P(XG,YG).



Bedenk bij de navolgende uiteenzetting dat de coördinaten (XG,YG) van een beeldschermpunt 'gemeten' worden ten opzichte van de linkerbovenhoek van het beeldscherm, de oorsprong van ons HRG-coördinatenstelsel.



Een punt, in beeldschermcoördinaten uitgedrukt, ligt dus 'hoger' dan een ander punt (bij dezelfde XG-waarde) als de YG-waarde groter is en ligt 'lager' als de YG-waarde kleiner is.

We gaan als volgt te werk. We gebruiken twee arrays (H1(XG) en H2(XG)) waarin we voor alle XG-coördinaten in ons vlak de respectievelijk kleinste en grootste YG-waarde bewaren. Vinden we voor een punt met een bepaalde XG-waarde een bijbehorende YG-waarde die òf kleiner is dan H1(XG) òf groter is dan H2(XG), dan is het punt zichtbaar en wordt H1(XG) of H2(XG) gelijk gemaakt aan deze

nieuwe kleinste of grootste YG-waarde. Geldt voor een bepaalde XG dat H1(XG)<YG<H2(XG), dan is het punt (XG,YG) niet zichtbaar. H1(XG) en H2(XG) zijn te zien als twee horizonnen.

Als voor een bepaald punt $P_1(x,y,z)$ uit het vlak het beeldpunt P1(XG,YG) berekend is, gaan we kijken of YG kleiner dan of gelijk aan H1(XG) is (YG<=H1(XG)). Is dit zo dan is het punt P1 op het beeldscherm zichtbaar. Nu zetten we de vlag F1 op 1 en we maken H1(XG) gelijk aan YG (H1(XG)=YG). Is dit niet het geval (YG> H1(XG)), dan is P1 onzichtbaar en blijft de vlag F1 op 0 staan.

Nu gaan we naar het volgende punt $P_2(x,y,z)$ uit het vlak (we verhogen x met dx). Opnieuw berekenen we het beeldpunt P2(XG,YG). Weer wordt gekeken of YG<=H1(XG) en zonodig wordt de vlag F2 op 1 gezet. De punten P1 en P2 worden alleen dan door een rechte lijn (lijntje) verbonden als beide vlaggen F1 en F2 de waarde 1 hebben, dus als F1xF2=1.

Ditzelfde doen we voor de tweede 'horizon' H2(XG). We kijken of YG>=H2(XG), ..., enzovoorts.

Omdat we bij een vaste y-waarde de x-waarde steeds met dx ophogen (bijvoorbeeld dx=3) en hiermee steeds twee buurpunten P1(XG,YG) en P2(XG,YG) berekenen slaan we als het ware een aantal punten over waarvoor geen H1(XG)- en H2(XG)-waarden berekend worden. Door lineaire interpolatie tussen de horizonwaarden (H1(XG) en H2(XG)) van twee buurpunten zouden we de horizonwaarden voor de hiertussenliggende punten kunnen berekenen. We hebben deze waarden wel nodig, omdat het kan voorkomen dat, als we y met dy verhogen, we XG-waarden vinden waarvoor nog geen H1(XG)- en H2(XG)-waarden berekend zijn.

Een programma dat de hierboven geschetste werkwijze zou volgen zou, in BASIC geschreven, veel te langzaam zijn. Om de tekensnelheid acceptabel te houden, dat wil zeggen niet langer dan 10 minuten tekenen voor één figuur, passen we de volgende vereenvoudigingen toe:

1. We berekenen alleen de onderste (voor de kijker de bovenste) horizon H1(XG).

2. We passen geen lineaire interpolatie toe bij het berekenen van de array-waarden in H1(XG). Alle beeldpunten in een interval ter lengte dx krijgen dezelfde H1(XG)-waarde.

Als we de stapgrootte dx maar klein genoeg kiezen (een goede waarde is dx=3), krijgen we ondanks deze twee simplificaties toch accep-

tabele tekeningen. (Bekijk de volgende tekeningen en oordeelt u zelf.) Het niet-tekenen van 'voor de kijker' onzichtbare punten komt tot stand door het volgende stukje programma:

```
F1=0:L=INT(XG/DX)
IG YG<=H(L) THEN F1=1:H(L)=YG
```

Laten we dit met een voorbeeld illustreren. Stel we berekenen van een punt P(x,y,z) in het ruimtelijke vlak de projectie op het beeldscherm. Dit gebeurt door eerst met x en y de z-waarde te berekenen. (In programma 29 krijgt Y in regel 260, X in regel 280 en Z in regel 1000 een waarde.) Als de x-, y- en z-waarde van een punt in het vlak berekend zijn, wordt dit punt op het beeldscherm geprojecteerd door het berekenen van de twee coördinaten XG en YG. (In programma 29 gebeurt dit in regel 290-330.) Stel nu dat als beeldpunt het punt (XG=40 en YG=126) berekend is. Het programma zal de vlag F1 op nul zetten (regel 330) en de waarde L=INT(XG/DX) berekenen. Voor XG=40 en DX=3 wordt deze waarde INT(40/3), dus L=13. Dit betekent in feite dat de drie punten met XG=39, XG=40 en XG=41 dezelfde horizonwaarde L=13 opleveren. Mocht nu eerder in H(13) als horizonwaarde de waarde 132 opgeslagen zijn, dan is YG<=H(L) (regel 340) waar, want 126<=132 is waar, en is het punt (40,126) zichtbaar. Op dit moment wordt de vlag F1 gezet en wordt YG=126 de nieuwe horizonwaarde bij L=13 (regel 340).

De regels

```
190 DIM H(255)
200 FOR L= 0 TO 255
210 :  H(L)=1000
220 NEXT L
```

zorgen ervoor dat, voordat het tekenen begint, het hele scherm 'zichtbaar' is. Dit is zo als de onderste horizon (voor de kijker de bovenste horizon) de onderrand van het beeldscherm is. De waarde 1000 geeft in feite een horizon die 'ver onder' het beeldscherm ligt. Op p.77 staat het programma dat alle voor ons onzichtbare punten ook niet tekent.

Met W=45°, K=0.5, A=3 en K1=80 krijgt u de grafiek van $z=e^{-(x^2+y^2)}$ zoals die een paar pagina's eerder is getekend; de niet-zichtbare punten die achter de hoed liggen zijn inderdaad niet te zien!

```
100 REM PROGRAMMA 29 GRAFIEK VAN Z=F(X,Y),HIDDEN LINES
110 PRINT CHR$(147)
120 INPUT "ALPHA IN GRADEN (45-135)"; W
130 INPUT "VERKLEININGSFACTOR (.5-.75)"; K
140 INPUT "RECHTERGRENS VOOR X(>0)"; A
150 INPUT "VERGROTINGSFACTOR (30-80)"; K1
160 U=160:V=100:H=0.5:RD=π/180
170 C=K*COS(W*RD) : S=K*SIN(W*RD)
180 DX=3 : DY=5 : AF=A/115
190 DIM H(255)
200 FOR L= 0 TO 255
210 :  H(L)=1000
220 NEXT L
230 PRINT CHR$(147)
240 HIRES 7,8
250 FOR YY=-115 TO 115 STEP DY
260 :  Y=YY*AF
270 :  FOR XX=-115 TO 115 STEP DX
280 :     X=XX*AF : GOSUB 1000
290 :     XG=INT(U+XX+C*YY+H)
300 :     YG=INT(V-S*YY-Z +H)
305 :     IF XG<0 THEN XG=0:IF XG>320 THEN XG=320
310 :     IF YG<0 OR YG>200 THEN PRINT "FOUTE K1":STOP
320 :     IF XX>-115 THEN 360
330 :     F1=0 : L=INT(XG/DX)
340 :     IF YG<=H(L) THEN F1=1 : H(L)=YG
350 :     X1=XG : Y1=YG : GOTO 410
360 :     F2=0 : L=INT(XG/DX)
370 :     IF YG<=H(L) THEN F2=1 : H(L)=YG
380 :     X2=XG : Y2=YG
390 :     IF F1*F2=1 THEN LINE X1,Y1,X2,Y2,1
400 :     X1=X2 : Y1=Y2 : F1=F2
410 :  NEXT XX
420 NEXT YY
430 GET A$ : IF A$="" THEN 430
440 END
1000 Z=K1*EXP(-X*X-Y*Y)
1100 RETURN
```

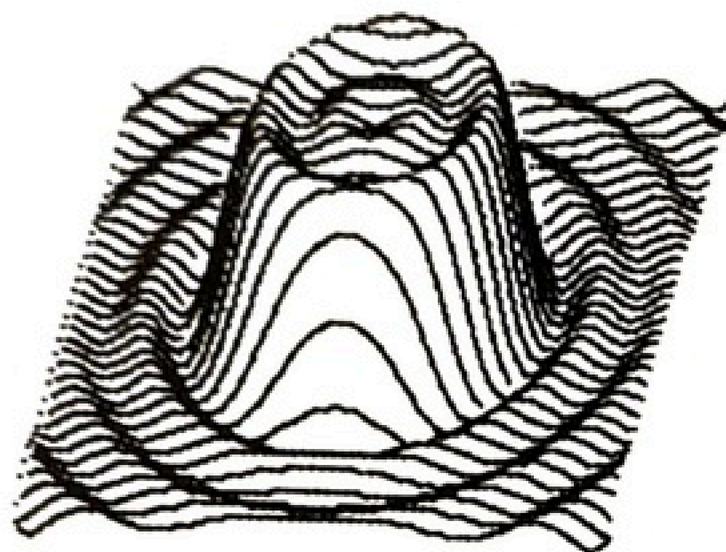
Hoe kleiner we DX en DY maken, des te gedetailleerde zal de figuur worden. Maar ook 'des te langer zal het tekenen duren'. De combinatie DX=3 en DY=5 is een goed compromis tussen de tekensnelheid en de mate van gedetailleerdheid.

Dit programma is een algemeen programma voor het tekenen van draaisymmetrische ruimtelijke vlakken. Als u andere figuren wilt maken, verander dan alleen de functie in de subroutine 1000 en kies mogelijk andere waarden voor de variabelen.

Nu volgt een aantal voorbeelden.

```
1000 R=SQR(X*X+Y*Y)*RD
1010 Z=K1*(COS(R)-COS(3*R)/3+COS(5*R)/5-COS(7*R)/7)
1100 RETURN
```

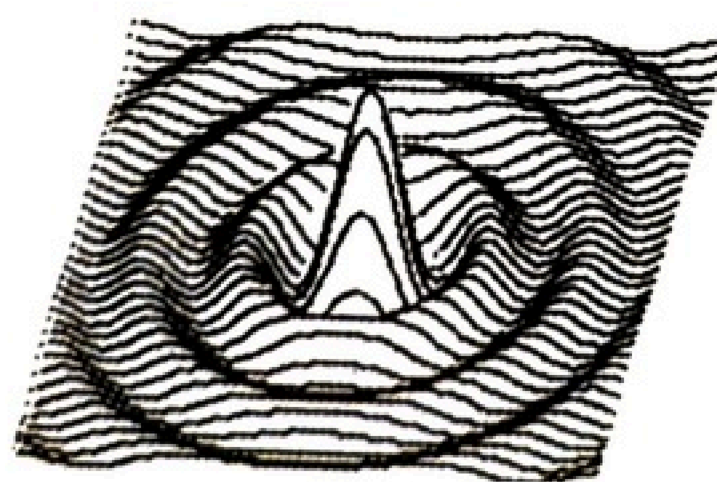
U kiest: W=45, K=0.5, A=180°, K1=35.



$$z = \cos(r) - \frac{\cos(3r)}{3} + \frac{\cos(5r)}{5} - \frac{\cos(7r)}{7}$$

```
1000 R=SQR(X*X+Y*Y)*RD
1010 IF R=0 THEN Z=K1 : RETURN
1020 Z=K1*SIN(R)/R
1100 RETURN
```

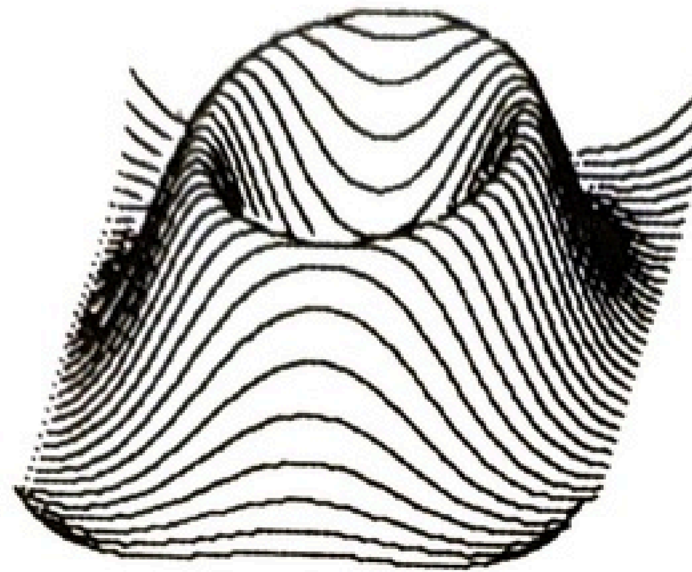
U kiest: W=45, K=0.5, A=108°, K1=50.



$$z = \frac{\sin r}{r}$$

```
1000 R=SQR(X*X+Y*Y)
1010 Z=K1*EXP(-COS(R/16))
1100 RETURN
```

U kiest: W=45, K=0.5, A=90, K1=30.

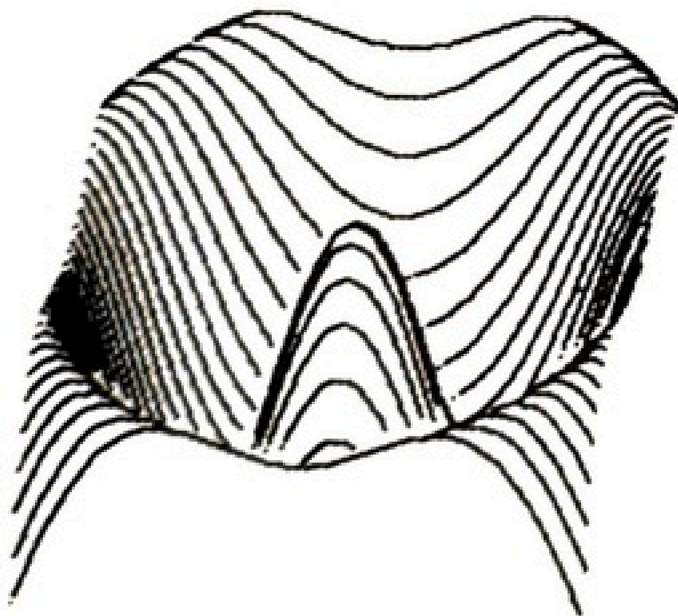


$z = \exp(-\cos(\frac{r}{16}))$

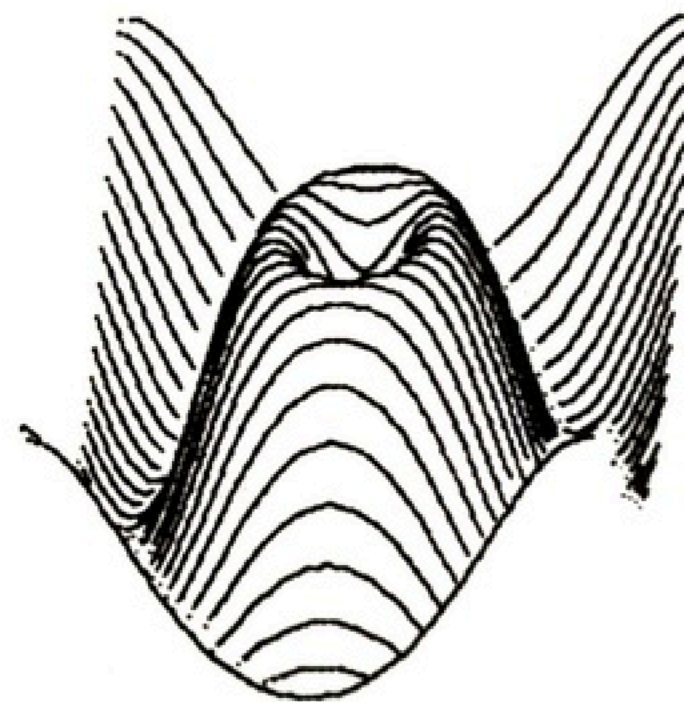
Voor de volgende twee tekeningen geldt:

```
1000 R=SQR(X*X+Y*Y)
1010 Z=K1*COS(R/16) RESPECT. Z=K1*SIN(R/16)
1100 RETURN
```

U kiest: W=45, K=0.5, A=90, K1=50.



$z = \cos(\frac{r}{16})$

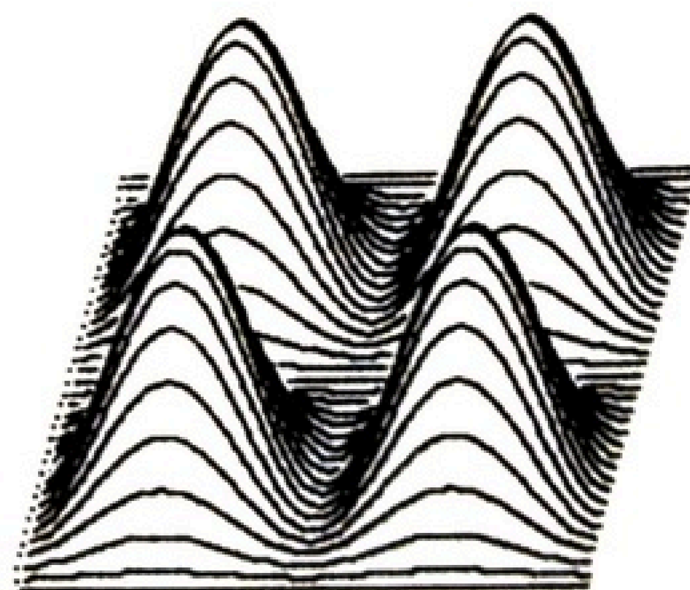


$z = \sin(\frac{r}{16})$

Met dit programma hebben we talrijke andere grafieken gemaakt. Elke keer als u een 'leuke functie' tegenkomt, kunt u meteen de daarbij horende draaisymmetrische figuur maken.

Tot slot volgt nog een programma voor het tekenen van een fraaie vier-toppige grafiek. U bent hem misschien wel eens in een computertijdschrift tegengekomen. Om de snelheid op te voeren is de subroutine 1000 in het hoofdprogramma opgenomen. U kunt dit natuurlijk ook in alle andere programma's doen.

```
100 REM PROGRAMMA 30 MOOIE FUNCTIE
110 PRINT CHR$(147)
120 INPUT "ALPHA IN GRADEN (45-135)"; W
130 INPUT "VERKLEININGSFACTOR (.5-.75)"; K
140 U=160:V=100:H=0.5:RD=π/180
150 C=K*COS(W*RD) : S=K*SIN(W*RD)
160 DX=3 : DY=5 : K1=15
170 DIM H(320)
180 FOR L= 0 TO 320
190 :  H(L)=1000
200 NEXT L
210 PRINT CHR$(147)
220 HIRES 7,8
230 FOR YY=-115 TO 115 STEP DY
240 :  M1=COS(YY*2*π/115-π)+1
250 :  FOR XX=-115 TO 115 STEP DX
260 :     M2=COS(XX*2*π/115-π)+1
270 :      Z=K1*M1*M2
280 :     XG=INT(U+XX+C*YY+H)
290 :     YG=INT(V-S*YY-Z +H)
300 :     IF XX>-115 THEN 340
310 :     F1=0 : L=INT(XG/DX)
320 :     IF YG<=H(L) THEN F1=1 : H(L)=YG
330 :     X1=XG : Y1=YG : GOTO 390
340 :     F2=0 : L=INT(XG/DX)
350 :     IF YG<=H(L) THEN F2=1 : H(L)=YG
360 :     X2=XG : Y2=YG
370 :     IF F1*F2=1 THEN LINE X1,Y1,X2,Y2,1
380 :     X1=X2 : Y1=Y2 : F1=F2
390 :  NEXT XX
400 NEXT YY
410 GET A$ : IF A$="" THEN 410
420 END
```



$$z = (\cos(\frac{2\pi x}{90} - \pi) + 1)(\cos(\frac{2\pi y}{90} - \pi) + 1)$$

Voor de bovenstaande tekening kiest u W=75°, K=0.50 en K1=20.

6 *Turtle-graphics en LOGO-simulatie*

LOGO is vooral door de turtle-graphics beroemd geworden. Er is geen andere programmeertaal waarmee zo eenvoudig zeer moeilijke grafische figuren gemaakt kunnen worden dan met LOGO. In dit hoofdstuk zullen we vijf LOGO-programma's in BASIC vertalen. Deze BASIC-programma's bevatten wel 25 tot 30 regels, terwijl LOGO hiervoor slechts 3 of 4 regels nodig heeft.

"Eerst LOGO, daarna andere programmeertalen"

is het motto van vele informatici en pedagogen die zich met de invoering van de informatica op scholen bezighouden.

Wanneer men deze stelling wil begrijpen, moet men niet alleen de taal LOGO en haar mogelijkheden, maar ook de omgeving waarin LOGO ontwikkeld werd goed kennen. Een korte historische terugblik lijkt daarom op zijn plaats.

Het schrijven van computerprogramma's is een intellectuele prestatie en een creatief proces. Aan de wijze waarop wij programma's ontwikkelen, herkennen wij direct onze manier van denken. Aan het Massachusetts Institute of Technology (MIT) heeft men al in het begin van de zestiger jaren een speciale programmeertaal ontwikkeld, met behulp waarvan men kunstmatige intelligentie ging bestuderen. Men noemde deze taal LISP, een afkorting van LISt-Processing.

In LISP werden programma's geschreven die een met kunstmatige zintuigen uitgeruste elektromechanische muis in staat stelden een uitweg uit ieder willekeurig doolhof te vinden. In LISP worden programma's ontwikkeld die de computer tot een medespeler met leervermogen maakt. Hoe meer spelletjes de computer tegen een menselijke tegenspeler speelt, des te beter wordt hij. Goede zetten van de tegenstander onthoudt hij en legt hij vast in zijn geheugen, terwijl hij de eigen slechte zetten voor toekomstige spelletjes uit zijn

geheugen wegveegt. Een 'afvalprodukt' van deze studies aan het MIT zijn de moderne schaakprogramma's.

LOGO is een hoog ontwikkeld dialect van de LISP-taal. Seymour Papert, leerling van de bekende onderzoeker Jean Piaget en medewerker aan het MIT, heeft in twaalf jaar tijd LOGO ontwikkeld. Jean Piaget heeft in zijn bekende boek *Hoe kinderen leren* de kinderlijke denkstructuren laten zien. Hieruit blijkt dat tekenen en knutselen tot de eerste creatieve handelingen van kinderen behoren. Wij zullen zien hoe LOGO deze bezigheden ondersteunt.

Amerikanen gaan door voor een volk dat graag experimenteert. Het was hun echter duidelijk, dat jonge kinderen, wij denken dan aan zes- tot dertienjarigen, niet in staat zijn een programmeertaal zoals BASIC, laat staan Pascal, te leren en algorithmen voor numerieke, niet-numerieke of grafische problemen te schrijven. Daarom heeft men een 'kinderlijke' taal gemaakt (waarachter echter een imponerende software schuilgaat), waarmee het kind met gemak tekeningen kan maken.

Wordt het LOGO-systeem ingeschakeld, tegenwoordig meestal een 64K-microcomputer met bijbehorende software, dan verschijnt in het midden van het beeldscherm een kleine driekhoek, waarvan de top naar boven wijst. De kinderen noemen die driehoek 'turtle', het Engelse woord voor schildpad. Met behulp van eenvoudige bevelen kan het kind de schildpad willekeurig over het beeldscherm laten rondlopen. En al naar gelang het wenselijk is laat de turtle wel of geen spoor na.

De volgende LOGO-opdrachten zijn beschikbaar:

FORWARD	100	= 100 passen vooruit
BACK	50	= 50 passen achteruit
RIGHT	90	= draaiing van 90° met de klok mee
LEFT	45	= draaiing van 45° tegen de klok in
PENUP		= haal pen van papier
PENDOWN		= zet pen op papier
HIDETURTLE		= haal Turtle van het beeldscherm

Normaal gesproken geldt de toestand "PENDOWN". Als we bijvoorbeeld de hoofdletter F op het scherm willen tekenen, kan dat in LOGO als volgt:

```
FORWARD 100  RIGHT 90  FORWARD 50
RIGHT 90  PENUP  FORWARD 50  PENDOWN
RIGHT 90  FORWARD 50
HIDETURTLE
```

Voor 100, 50, 90 en 45 kunnen ook andere waarden gekozen worden.

Omdat LOGO een vertolkend systeem is, wordt elke opdracht (na het geven van RETURN) direct uitgevoerd. Zo op het oog lijkt het slechts leuk speelgoed! Laten we nu eens een LOGO-programma bekijken; liever spreken we van een procedure:

```
TO VIERKANT
REPEAT 4 (FORWARD 75  RIGHT 90)
END
```

Het LOGO-systeem weet dat tussen TO en END een procedure (een stuk programma) gedefinieerd wordt. De procedure tussen TO en END (in ons voorbeeld VIERKANT genoemd) maken we zelf. Zouden we bovenstaande procedure intoetsen, dan zal op het beeldscherm een vierkant met zijden ter lengte 75 getekend worden. De programmeurs onder u kennen vast de opdracht REPEAT waarmee een herhalingsstructuur geprogrammeerd kan worden.

Als we de procedure VIERKANT in het LOGO-systeem ingevoerd hebben, kunnen we hier ook gebruik van maken. LOGO onthoudt alle procedures die wij zelf invoeren. We kunnen dan ook later nieuwe procedures ontwikkelen waarin we gebruik maken van eerder ingevoerde procedures (zoals VIERKANT). Nu volgt een procedure met een parameter:

```
TO VIERKANT:ZIJDE
REPEAT 4 (FORWARD:ZIJDE RIGHT90)
END
```

Toetsen we nu VIERKANT 150 <RETURN> in, dan zal LOGO een vierkant met zijden van 150 tekenen. Niets verhindert u een procedure in te bedden in een volgende procedure en die weer in een volgende en die weer ....., enzovoorts. Alleen de beschikbare geheugenruimte is hierbij een remmende factor. Bekijk het volgende LOGO-programma.

LOGO-programma nr. 1

```
TO VIERKANTPATROON
REPEAT 8 (FORWARD 20  LEFT 45  VIERKANT 75)
END
```

Dit eenvoudige drieregelige LOGO-programma tekent een patroon van acht vierkanten (zie p. 87). We zullen dit programma straks in BASIC vertalen. Het zal blijken dat hiervoor enige wiskundige en programmeertechnische kennis nodig is. Het LOGO-programma kan door een leerling van de lagere school gemaakt worden, terwijl het

BASIC-programma dat deze acht vierkanten tekent slechts door scholieren van de hoogste klassen van het voortgezet onderwijs kan worden gemaakt. Het wordt nog moeilijker als we intikken:

```
TO STROOK:AANTAL
REPEAT:AANTAL (FORWARD 100  VIERKANTPATROON)
END
```

Tikken we vervolgens in STROOK 5 <RETURN> dan maken we heel eenvoudig een fraai ornament. Probeer dit maar eens in BASIC of Pascal. LOGO wordt pas echt interessant als we van de mogelijkheid gebruik maken dat een procedure zichzelf aanroept (recursiviteit). De turtle-graphics berusten op het principe dat we een procedure parameters geven en dat deze procedure zichzelf steeds met andere parameterwaarden aanroept. Bekijk de volgende procedure:

```
TO VEELHOEK:ZIJDE:HOEK
FORWARD:ZIJDE  LEFT:HOEK
VEELHOEK:ZIJDE:HOEK   ←   hier roept de procedure
END                       zichzelf aan
```

Als we nu VEELHOEK 200 144 intikken, zal het LOGO-systeem een regelmatige vijfpuntige ster tekenen. We zullen echter op de STOP-toets moeten drukken om het tekenen te stoppen. Brengen we in deze procedure een stopmechanisme aan en laten we de procedure steeds de waarde van ZIJDE veranderen, dan krijgen we de welhaast bekendste LOGO-procedure:

LOGO-programma nr. 2

```
TO TURTLE:ZIJDE:GROEI:HOEK
FORWARD:ZIJDE  LEFT:HOEK
MAKE:ZIJDE:ZIJDE+:GROEI
IF:ZIJDE>200 THEN STOP
TURTLE:ZIJDE:GROEI:HOEK
END
```

Als er een dubbele-punt voor een LOGO-woord staat, weet het LOGO-ssyteem dat het om een naam van een variabele gaat en niet om de naam van een procedure of een LOGO-opdracht. In de bovenstaande LOGO-procedure zien we hoe de procedure zichzelf steeds met een nieuwe waarde voor ZIJDE aanroept. Ook dit programma vertalen we in BASIC. Hiermee kunt u elke denkbare rechtlijnige turtle-grafiek maken. U hoeft alleen de waarden van de invoerparameters (ZIJDE, GROEI en HOEK) te veranderen.

Het derde LOGO-programma tekent een reeks vierkanten in spiraalvorm. Het zijn bekende figuren in het 'land van de computergraphics'.

LOGO-programma nr.3

```
TO VIERKANTSPIRAAL:ZIJDE:HOEK
IF:ZIJDE>150 THEN STOP
VIERKANT:ZIJDE  LEFT:HOEK
VIERKANTSPIRAAL:ZIJDE+4:HOEK
END
```

Ook dit LOGO-programma vertalen we in BASIC. Met deze drie programma's hebt u niet alleen met LOGO maar ook met de 'turtlegraphics' kennis gemaakt. Wilt u meer weten van deze graphicswereld, dan wijzen wij op het boek *Turtle Geometry: The Computer as a Medium for Exploring Mathematics* van Harold Abelson en Andrea di Sessa, uitgegeven door Cambridge, MA:MIT Press, 1981. Liefhebbers van wiskunde, informatica en LOGO vinden in dit boek een schat aan informatie.

Natuurlijk zijn we in LOGO niet gebonden aan 'rechte lijnen'. In LOGO hebben we de beschikking over alle wiskundige functies en bewerkingen. Alle programma's uit dit boek kunnen in LOGO geschreven worden. Zij zouden dan vast korter, overzichtelijker en begrijpelijker geweest zijn. In LOGO kunnen we heel eenvoudig met lijsten en tabellen werken. In LOGO hoeven we de variabelen niet te declareren. Achter een naam in LOGO kan gewoon een getal, een string, een n-dimensionaal veld en nog veel meer schuil gaan. Ook kan het de naam van een procedure zijn. We hoeven ons niet te bekommeren om het reserveren van geheugenruimte (DIM-opdracht in BASIC), noch om het aangeven van het type (real, integer, character, boolean, enz.); het LOGO-systeem regelt dit zelf.

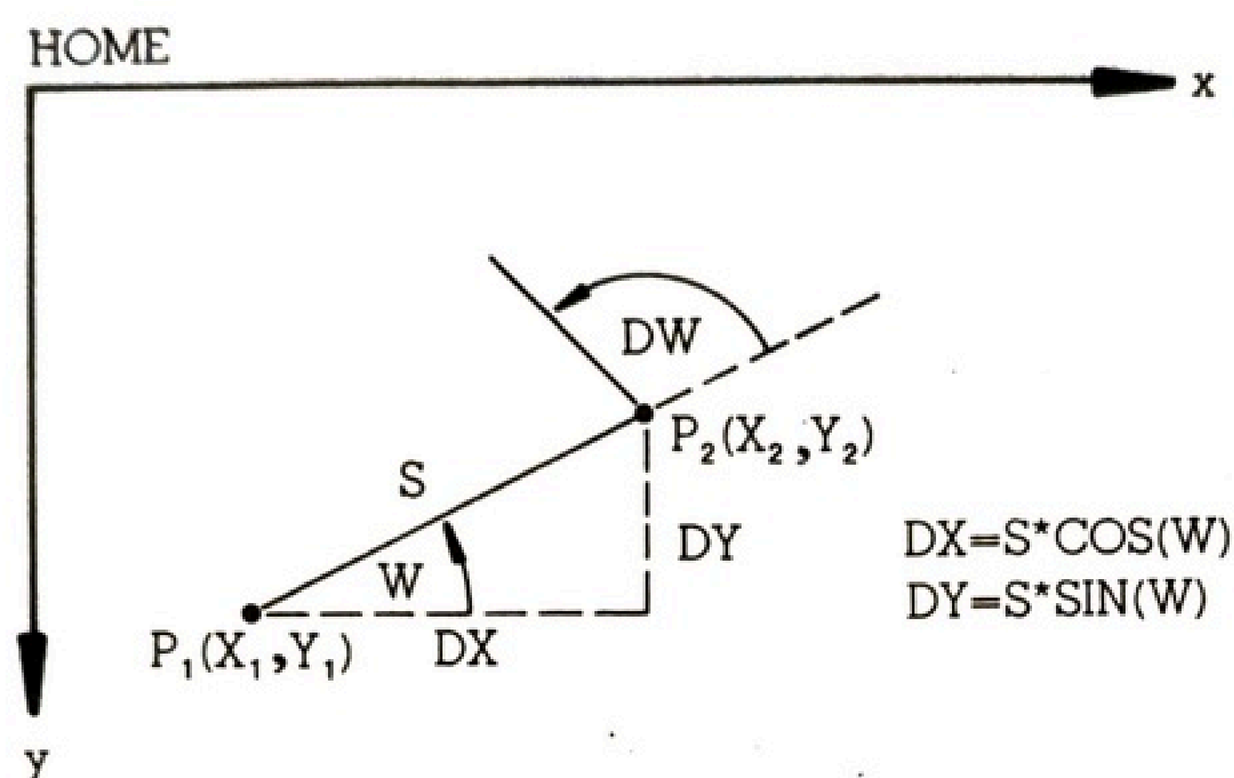
Wie in LOGO programmeert leert automatisch netjes programmeren. Begrippen als procedure, recursie, herhaling, toekenning worden op een natuurlijke manier aangeleerd. Met een paar opdrachten is een kind al in staat ingewikkelde figuren te tekenen. Met LOGO degraderen we de computer niet tot een programmeerbare zakrekenmachine, maar halen we er alles uit wat erin zit.

Vertaling LOGO-programma nr.1 in BASIC

De kern van elk LOGO-graphics-programma wordt gevormd door twee opdrachten:

```
FORWARD (resp. BACK)    :ZIJDE
LEFT (resp. RIGHT)      :HOEK
```

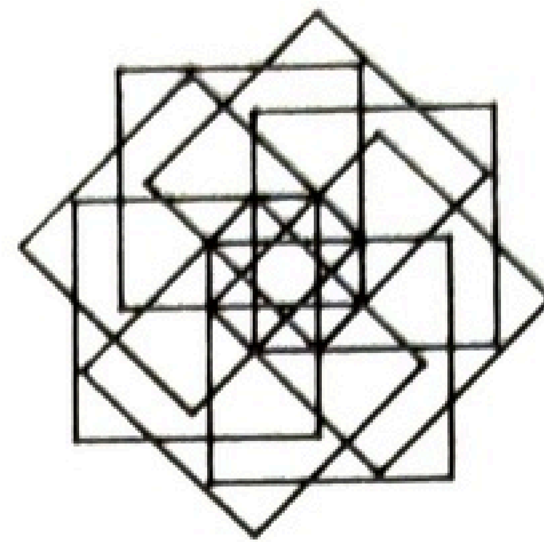
Hoe vertalen we deze opdrachten in BASIC? Bekijk hiertoe de volgende figuur.



De PEN staat in punt $P_1(x_1,y_1)$ en moet in de richting van hoek W over een afstand S verplaatst worden. In $P_2(x_2,y_2)$ aangekomen moet de PEN (eigenlijk de turtle) een hoek van DW° linksom maken. De volgende BASIC-opdrachten voeren dit uit:

```
INPUT "X1,Y1,W" ; X1,Y1,W
H=0.5 : RD=π/180 : W1=W*RD
X2=INT(X1+S*COS(W1)+H)
Y2=INT(Y1-S*SIN(W1)+H)
LINE X1,Y1,X2,Y2,1 : X1=X2 : Y1=Y2
W=W+DW : IF W>=360 THEN W=W-360
W1=W*RD
```

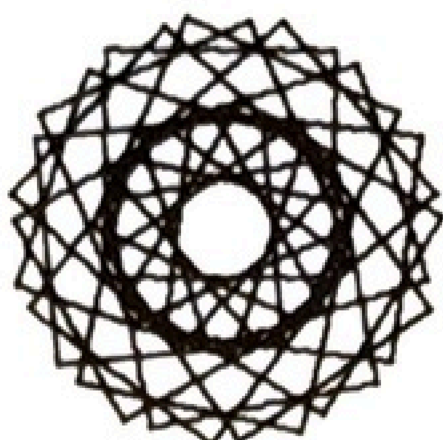
Deze opdrachten komt u in alle volgende BASIC-programma's tegen. Meer theorie is niet nodig! De programma's moeten met bovenstaande informatie gelezen kunnen worden.



Experimenteer met programma 31 LOGO-1 vierkantpatroon. Met W=90°, S1=20, DW=45, S2=75 en N=8 krijgen we de bovenstaande tekening.

```
100 REM PROGRAMMA 31 VIERKANTPATROON(LOGO-1)
110 INPUT "COORDINATEN STARTPUNT";X1,Y1
120 INPUT "BEGINRICHTING (GRADEN)"; W
130 INPUT "VERPLAATSING           "; S1
140 INPUT "DRAAIHOEK,LINKSOM      "; DW
150 INPUT "ZIJDE VIERKANT         "; S2
160 INPUT "AANTAL VIERKANTEN      "; N
170 H=0.5 : RD=π/180 : W1=W*RD
180 PRINT CHR$(147)
190 HIRES 0,1
200 FOR J=1  TO N
210 :  X2=INT(X1+S1*COS(W1)+H)
220 :  Y2=INT(Y1-S1*SIN(W1)+H)
230 :  LINE X1,Y1,X2,Y2,1:X1=X2:Y1=Y2
240 :  AX=X1 : AY=Y1
250 :  W=W+DW : IF W>360 THEN W=W-360
260 :  W1=W*RD
261 :  FOR K=0 TO 2
270 :     X2=INT(X1+S2*COS(W1+K/2*π)+H)
280 :     Y2=INT(Y1-S2*SIN(W1+K/2*π)+H)
290 :     LINE X1,Y1,X2,Y2,1:X1=X2:Y1=Y2
291 :  NEXT K
360 :  LINE X1,Y1,AX,AY,1:X1=AX:Y1=AY
370 NEXT J
380 GET A$ : IF A$="" THEN 380
390 END
```

W=90, S1=30, DW=48, S2=70 en N=15 geeft de onderstaande tekening. Als DW een deler is van 360, is N gelijk aan 360/DW. Kiest u voor DW een willekeurige waarde, neem dan voor N een groot getal, bijvoorbeeld 100, en breek het tekenen met de stoptoets af. De 'kinderen' doen dat in LOGO ook op deze manier.



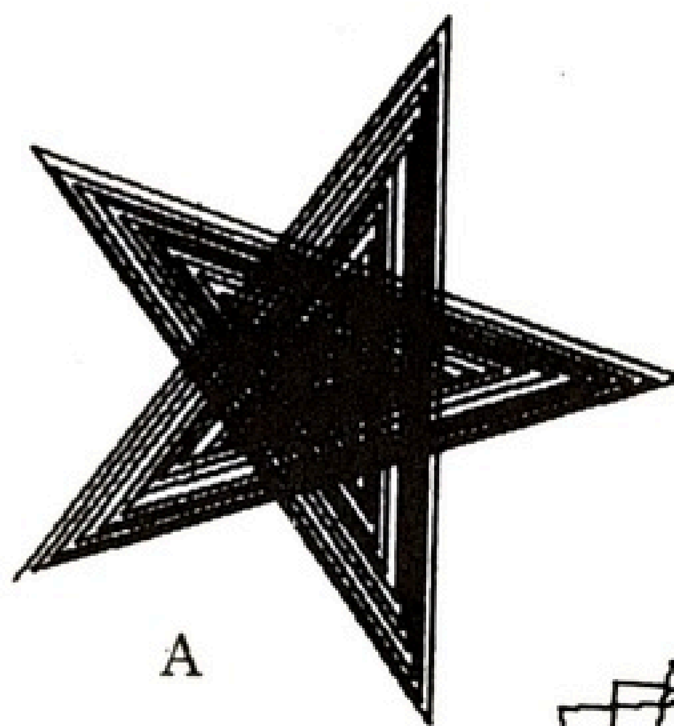
Vertaling LOGO-programma nr.2 in BASIC

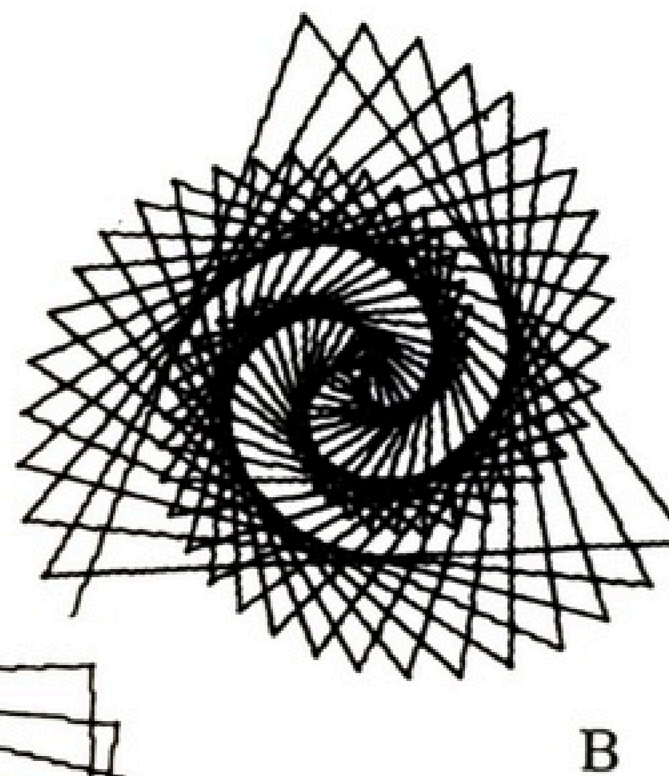
Het programma lijkt sterk op het vorige en behoeft daarom geen commentaar.

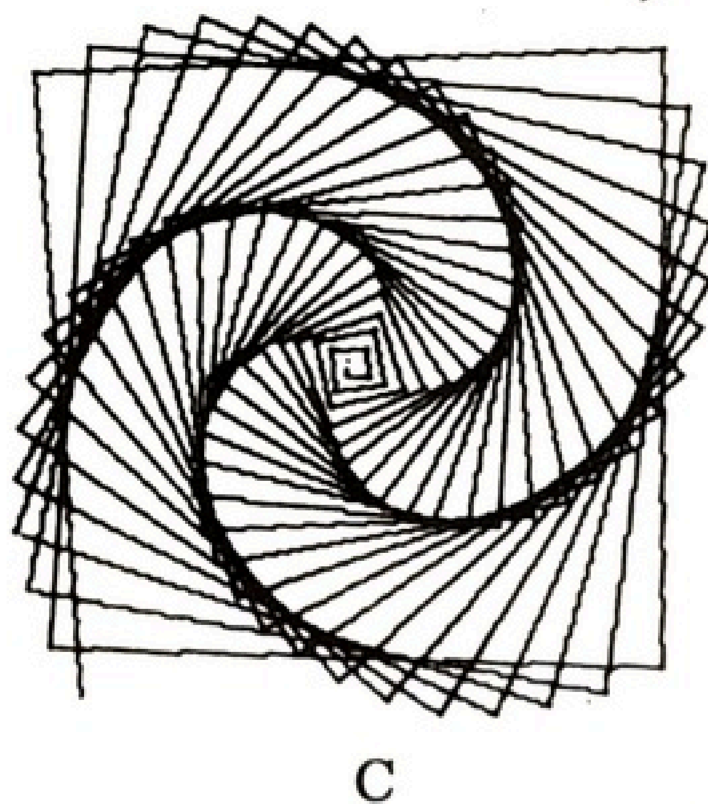
```
100 REM PROGRAMMA 32 TURTLE-GRAFIEK(LOGO-2)
110 PRINT CHR$147)
120 INPUT "COORDINATEN STARTPUNT";X1,Y1
130 INPUT "BEGINRICHTING (GRADEN)"; W
140 INPUT "VERPLAATSING          "; S
150 INPUT "DRAAIHOEK,LINKSOM     "; DW
160 INPUT "TOENAME ZIJDE         "; DS
170 H=0.5 : RD=π/180 : W1=W*RD
180 PRINT CHR$(147)
190 HIRES 0,1
200 X2=INT(X1+S*COS(W1)+H)
210 Y2=INT(Y1-S*SIN(W1)+H)
220 IF X2<0 OR X2>320 OR Y2<0 OR Y2>200 THEN 280
230 LINE X1,Y1,X2,Y2,1
240 X1=X2 : Y1=Y2
250 W=W+DW : IF W>=360 THEN W=W-360
260 W1=W*RD : S=S+DS
270 GOTO 200
280 GET A$ : IF A$="" THEN 280
290 END
```

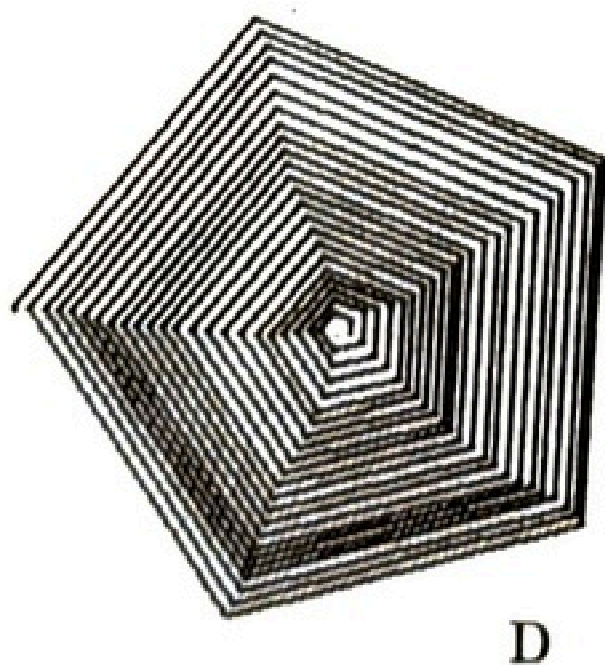
Het is handig de ingevoerde waarden na het tekenen op het scherm of op een printer te laten afdrukken. In de volgende tabel zien we de diverse waarden voor de daaronderstaande tekeningen.

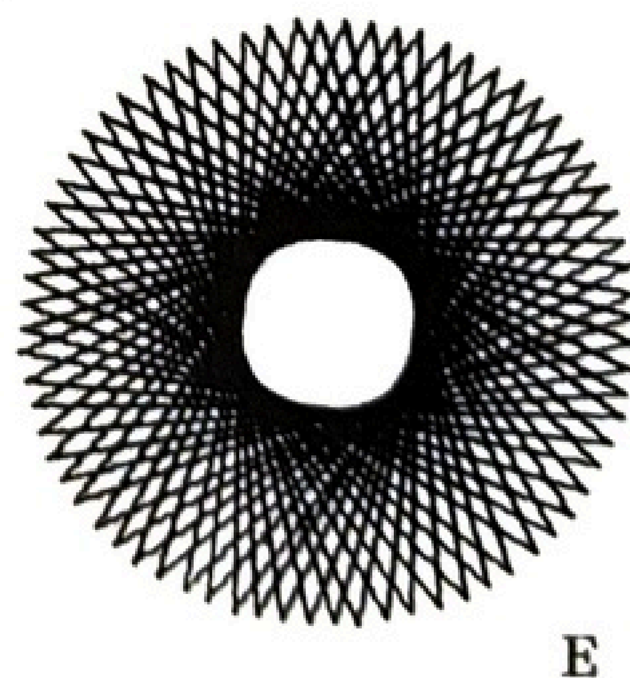
Tekening	X1	Y1	W	S	DW	DS
A	160	100	90	5	144	3
B	160	100	90	5	123	2
C	160	100	90	-2	92	2
D	160	100	90	5	72	1
E	160	170	90	160	145	0

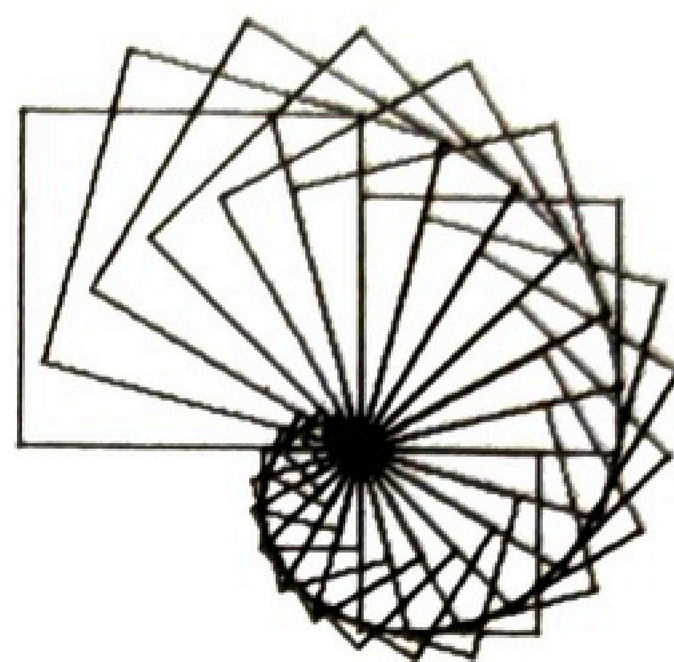

A


B


C


D


E



Vertaling LOGO-programma nr.3 in BASIC

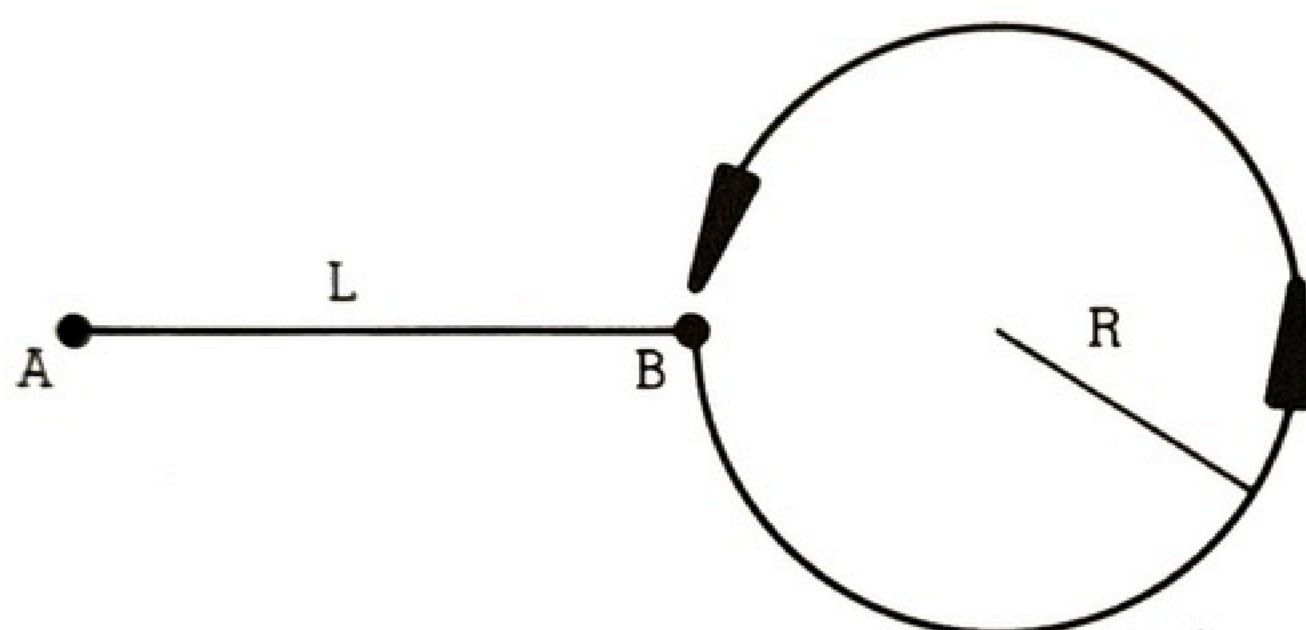
```
100 REM PROGRAMMA 33 VIERKANTSPIRAAL(LOGO-3)
110 INPUT "COORDINATEN STARTPUNT";X1,Y1
120 INPUT "BEGINRICHTING (GRADEN)"; W
130 INPUT "ZIJDE BEGINVIERKANT   "; S
140 INPUT "DRAAIHOEK,LINKSOM     "; DW
150 INPUT "TOENAME ZIJDE         "; DS
160 H=0.5 : RD=π/180 : W1=W*RD
170 PRINT CHR$(147)
180 HIRES 0,1
190 AX=X1 : AY=Y1
200 FOR FASE=0 TO π STEP π/2
210 :  GOSUB 1000
220 :  IF VLAG=1 THEN STOP
230 :  LINE X1,Y1,X2,Y2,1:X1=X2:Y1=Y2
240 NEXT FASE
250 LINE X1,Y1,AX,AY,1:X1=AX:Y1=AY
260 W=W+DW : IF W>=360 THEN W=W-360
270 W1=W*RD : S=S+DS
280 GOTO 200
290 GET A$ : IF A$="" THEN 290
300 END
305 :
1000 X2=INT(X1+S*COS(W1+FASE)+H)
1010 Y2=INT(Y1-S*SIN(W1+FASE)+H)
1020 IF X2<0 OR X2>320 OR Y2<0 OR Y2>200 THEN VLAG=1
1030 RETURN
```

Voor de boven het programma staande tekening kozen we X1=160, V1=100, W=90, S=5, DW=15 en DS=3.5.

Vergelijk deze drie BASIC-programma's eens met de overeenkomstige LOGO-programma's. In BASIC moeten we zelf alle graphic-software schrijven, terwijl deze in LOGO als machine-routines aanwezig is.

Het 4e en 5e LOGO-programma

Tot slot van dit hoofdstuk geven we nog twee BASIC-programma's waarmee de turtle behalve rechte wegen ook kromme wegen kan bewandelen. Dit voegt een nieuw element aan de turtle-graphics toe. De LOGO-structuur zullen we stapsgewijs verfijnen. Als uitgangspunt nemen we de onderstaande figuur. Deze bestaat uit een lijnstuk L, met daaraan vast een cirkel met straal R. We noemen deze figuur voor het gemak even 'de steelpan'.



Als de turtle de steelpan tekent, begint hij in A. Hij legt vervolgens een afstand L in positieve x-richting af en komt in B. Hier draait hij 90° met de klok mee en legt daarna de omtrek van de cirkel af tot hij weer in B uitkomt. Daar draait hij zich 90° tegen de klok in en kijkt dan rechtvooruit naar het punt A. Met deze steelpan kunnen we twee soorten tekeningen maken. Laten we deze mogelijkheden bekijken:

LOGO-programma 4

```
TO CIRKELFIGUUR-1:LIJNSTUK:STRAAL:HOEK
STEELPAN:LIJNSTUK:STRAAL
LEFT:HOEK
CIRKELFIGUUR-1:LIJNSTUK:STRAAL:HOEK
END
```

LOGO-programma 5

```
TO CIRKELFIGUUR-2:LIJNSTUK:STRAAL:HOEK
REPEAT 4 (STEELPAN:LIJNSTUK:STRAAL  LEFT 90)
LEFT:HOEK
CIRKELFIGUUR-2:LIJNSTUK:STRAAL:HOEK
END
```

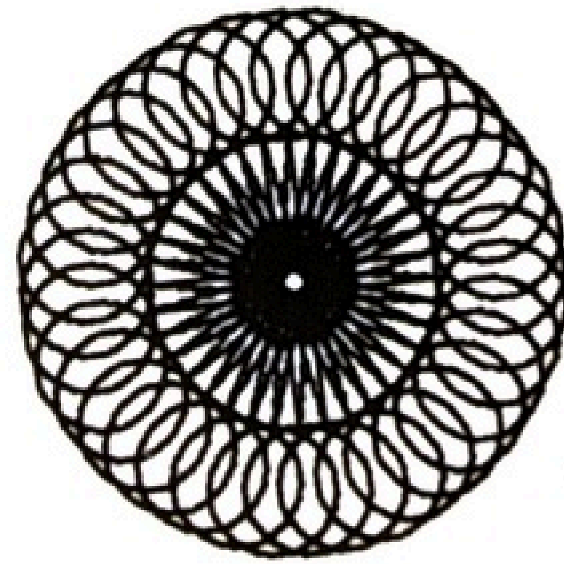
Deze twee LOGO-programma's kunnen nog niet uitgevoerd worden, want de procedure STEELPAN is nog niet in het LOGO-systeem gedefinieerd. We gaan de structuur verfijnen:

```
TO STEELPAN:LIJNSTUK:STRAAL
FORWARD:LIJNSTUK  RIGHT 90
CIRKEL:STRAAL
LEFT 90
END
```

Ook in deze procedure zien we een nog niet gedefinieerde procedure, CIRKEL, opduiken. Om de turtle een cirkel te kunnen laten maken zullen we deze procedure CIRKEL moeten maken. Veel LOGO-versies bevatten reeds zo'n voorgedefinieerde CIRKEL-procedure. LOGO-versies die hierin niet voorzien zullen dezelfde trigonometrische functies moeten gebruiken, zoals wij die in het volgende programma 34 hebben geprogrammeerd. In de volgende twee programma's zien we dat het tekenen van een cirkel geprogrammeerd is met een FOR-NEXT-lus waarin de hoek van W1-π tot WP+π loopt (programma 34, regels 260-300). We kunnen deze lus natuurlijk vervangen door de opdrachten

```
U = INT(AX+R*COS(W1)+π)
V = INT(AY-R*SIN(W1)+π)
CIRCLE U,V,R,R,1
```

als we over Simon's BASIC beschikken.



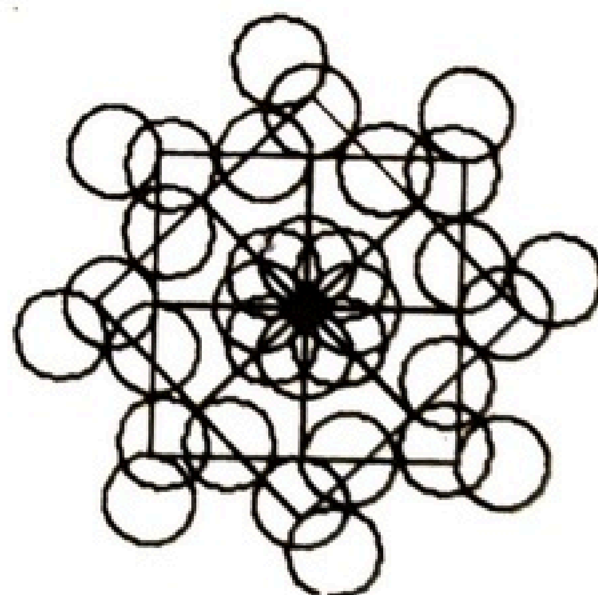
Vertaling van LOGO-programma nr.4 in BASIC

```
100 REM PROGRAMMA 34 CIRKELFIGUUR-1(LOGO-4)
105 PRINT CHR$(147)
110 INPUT "COORDINATEN STARTPUNT";X1,Y1
120 INPUT "BEGINRICHTING (GRADEN)"; W
130 INPUT "LENGTE VAN DE STEEL   "; L
140 INPUT "STRAAL VAN DE CIRKEL  "; R
150 INPUT "DRAAIHOEK,LINKSOM     "; DW
160 H=0.5 : RD=π/180 : W1=W*RD
170 PRINT CHR$(147)
180 HIRES 0,1
190 XX=X1 : YY=Y1
200 X2=INT(X1+L*COS(W1)+H)
210 Y2=INT(Y1-L*SIN(W1)+H)
220 LINE X1,Y1,X2,Y2,1
230 AX=X2 : AY=Y2
235 REM CIRKEL M(U,V) EN STRAAL R
236 U=INT(AX+R*COS(W1)+H)
240 V=INT(AY-R*SIN(W1)+H)
250 X1=AX : Y1=AY
260 FOR WW=(W1-π) TO (W1+π) STEP π/32
270 :  X2=INT(U+R*COS(WW)+H)
280 :  Y2=INT(V-R*SIN(WW)+H)
290 :  LINE X1,Y1,X2,Y2,1:X1=X2:Y1=Y2
300 NEXT WW
310 W=W+DW : IF W>=360 THEN W=W-360
320 W1=W*RD : X1=AX : Y1=AY
330 IF X1=XX AND Y1=YY THEN 350
340 GOTO  200
350 GET A$ : IF A$="" THEN 350
360 END
```

De bovenstaande tekening krijgen we door X1=160, Y1=140, W=90, L=90, R=20 en DW=170 te kiezen.

Vertaling LOGO-programma nr.5 in BASIC

```
100 REM PROGRAMMA 35 CIRKELFIGUUR-2(LOGO-5)
105 PRINT CHR$(147)
110 INPUT "COORDINATEN STARTPUNT";X1,Y1
120 INPUT "BEGINRICHTING (GRADEN)"; W
130 INPUT "LENGTE VAN DE STEEL    "; L
140 INPUT "STRAAL VAN DE CIRKEL   "; R
150 INPUT "DRAAIHOEK,LINKSOM      "; DW
160 H=0.5 : RD=π/180 : W1=W*RD
170 PRINT CHR$(147)
180 HIRES 0,1
190 FOR J=1 TO 4
200 :  X2=INT(X1+L*COS(W1)+H)
210 :  Y2=INT(Y1-L*SIN(W1)+H)
220 :  LINE X1,Y1,X2,Y2,1
230 :  AX=X2 : AY=Y2
235 :  REM CIRKEL M(U,V) EN STRAAL R
236 :  U=INT(AX+R*COS(W1)+H)
240 :  V=INT(AY-R*SIN(W1)+H)
250 :  X1=AX : Y1=AY
260 :  FOR WW=(W1-π) TO (W1+π) STEP π/32
270 :     X2=INT(U+R*COS(WW)+H)
280 :     Y2=INT(V-R*SIN(WW)+H)
290 :     LINE X1,Y1,X2,Y2,1:X1=X2:Y1=Y2
300 :  NEXT WW
310 :  W=W+90 : IF W>=360 THEN W=W-360
320 :  W1=W*RD : X1=AX : Y1=AY
330 NEXT J
340 W=W+DW : IF W>=360 THEN W=W-360
350 W1=W*RD
360 GOTO 190
370 END
```



Voor deze tekening geldt: X1=160, Y1=100, W=0, L=50, R=10 en DW=45.
Met andere waarden voor deze variabelen kunt u de mooiste turtleplaatjes maken.

7 *Educatieve toepassings-programma's*

De voorgaande 35 grafische programma's zijn geen toepassingsprogramma's. Ze waren bedoeld om een overzicht te geven van de mogelijkheden van graphics met hoog oplossend vermogen. In dit hoofdstuk zullen we vijf praktijkgerichte grafische programma's presenteren. Deze programma's zijn:

1. Tekenen van een landkaart
2. Maken van een histogram (een stavengrafiek)
3. Demonstratieprogramma voor de breking van lichtstralen
4. Demonstratieprogramma voor de 'speldenworp' van Buffon
5. Prooi-roofdierpopulaties

1. Tekenen van een landkaart

Educatieve programma's over topografie gebruiken dikwijls landkaarten van een land of van een werelddeel die door de computer getekend worden. We zullen laten zien hoe de computer zo'n landkaart kan tekenen. We gaan de landkaart (althans de grensomtrek) van Zwitserland tekenen. Waarom Zwitserland? Wel, in de eerste plaats omdat de auteur dit gekozen heeft en in de tweede plaats omdat Zwitserland geen eilanden heeft. Het tekenen van de contouren van Nederland gaat in principe net zo, alleen kosten al die eilanden een hoop extra werk, vandaar!

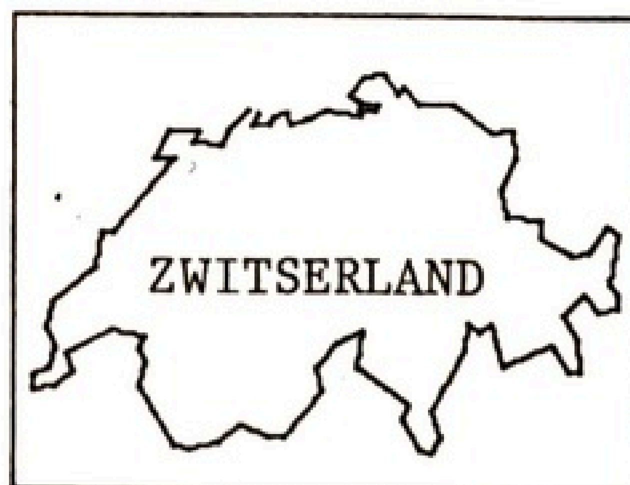
Hoe tekenen we nu de omtrek van een bepaald land? Heel eenvoudig! We pakken gewoon een atlas, een stukje overtrekpapier, een potlood, een lineaal en millimeterpapier. We kiezen in de atlas een land of werelddeel uit en trekken het op overtrekpapier over. Daarna leggen we het overgetrokken plaatje op millimeterpapier, waar we van te voren een coördinatenstelsel op getekend hebben (een x- en een y-as). We bepalen nu van een (groot) aantal punten op de omtrek van het land (of werelddeel) de coördinaten (x,y) en schrijven deze op. Als we ons op het beeldscherm ook een coördinatenstelsel voorstellen en als we hierin de punten, waarvan we de coör-

dinaten in een programma opnemen, met elkaar laten verbinden, dan ontstaat een 'landkaart' op het scherm.

In het onderstaande programma, dat 'de kaart' van Zwitserland tekent, is een 'echte' kaart gebruikt met een schaal van 1 : 2.000.000. Om een enigszins natuurgetrouwe weergave te verkrijgen zijn de coördinaten van 90 grenspunten berekend. De coördinaten van deze punten, die opgegeven zijn in millimeters ten opzichte van de oorsprong van het gekozen coördinatenstelsel, zijn in DATA-regels opgenomen. Als het programma het coördinatenpaar (0,0) leest, weet het dat de 'kaart' af is.

De x-coördinaten liggen tussen 6 en 177. Om deze naar het hogeresolutiebereik 0-220 te transformeren vermenigvuldigen wij zowel de x- als de y-coördinaten met de factor K = 200/170. Hierdoor blijft nog wat ruimte over om een kader rond de kaart te tekenen. De werking van het programma zal hiermee duidelijk zijn.

We hebben op deze manier ook een kaart van Europa en zelfs een wereldkaart getekend. Het probleem met de eilanden hebben we als volgt opgelost. Als het programma in een DATA-regel negatieve x- en y-coördinaten leest, weet het programma dat dit punt niet met het vorige verbonden moet worden en dat dit punt dus het begin is van een apart stukje 'land'.



```
100 REM PROGRAMMA 36 KAART VAN ZWITSERLAND
110 PRINT CHR$(147)
120 HIRES 0,1
130 K=1.5:H=0.5:U=23:V=200
140 READ X,Y
150 X1=INT(U+K*X+H):Y1=INT(V-K*Y+H)
160 :  X2=INT(U+K*X+H):Y2=INT(V-K*Y+H)
170 :  LINE X1,Y1,X2,Y2,1
180 :  X1=X2 : Y1=Y2
190 :  READ X,Y
200 IF X<> 0 THEN 160
210 GET A$ : IF A$="" THEN 210
220 END
230 :
300 DATA 69,108,71,107,70,104,75,104,75,104,76,106
310 DATA 80,107,81,104,86,105,91,108,94,107
320 DATA 101,106,100,108,105,108,106,110,101,110
330 DATA 98,112,102,117,108,118,112,112,114,115
340 DATA 118,110,128,110,139,102,145,103,146,95
350 DATA 142,86,144,78,154,77,154,77,154,72,163,67
360 DATA 168,68,173,76,177,73,174,59,177,56
370 DATA 177,52,171,51,167,56,161,50,165,43
380 DATA 166,34,162,34,157,42,143,36,139,48
390 DATA 136,45,133,48,132,40,122,23,125,15
400 DATA 122,11,119,12,114,20,116,25,100,35
410 DATA 102,45,94,42,88,35,90,28,79,15
420 DATA 75,15,66,19,60,14,52,12,48,13
430 DATA 37,29,39,36,37,40,39,43,29,45
440 DATA 16,38,18,33,13,29,6,28,6,32
450 DATA 11,34,13,40,10,45,12,53,25,63
460 DATA 26,73,30,73,48,94,42,94,46,102
470 DATA 54,102,53,99,61,98,63,102,69,108
480 DATA 0,0
```

Wat denkt u dat u ziet als u de volgende regels aan het programma toevoegt?

```
120 HIRES 1,0

214 IF A$ = "P" THEN PAINT 160,100,1
215 IF A$ = "V" THEN U=160:V=100:K=1:  GOTO 140
220 GET A$ : IF A$="" THEN 220
230 END

490 DATA -40, 20,-10, 20,-10, 50, 10, 50
500 DATA  10, 20, 40, 20, 40,  0, 10,  0
510 DATA  10,-30,-10,-30,-10,  0,-40,  0
520 DATA -40, 20,0,0
```

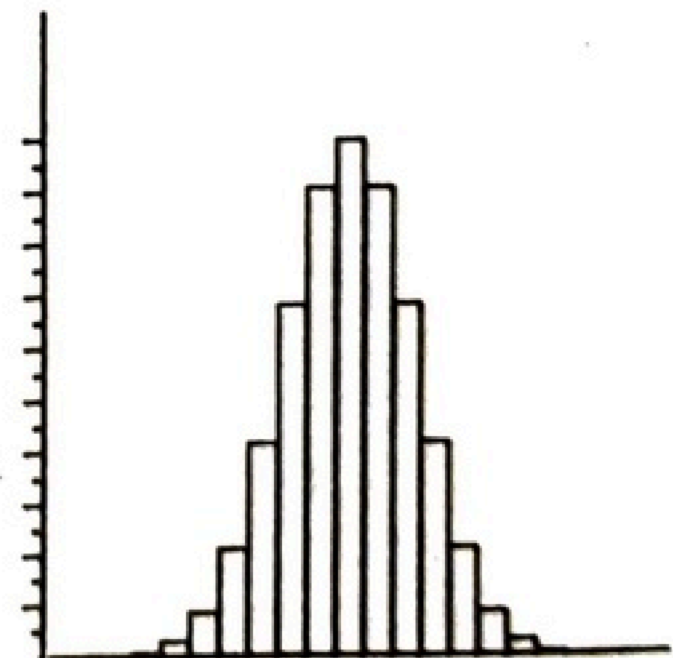
2. Maken van een histogram

Vaak willen we een aantal waarnemingen grafisch in een stavendiagram of histogram weergeven. Hiertoe dient het volgende programma. Dit programma tekent een horizontale en een verticale as. De verticale as wordt in stukjes van acht schermpuntjes verdeeld. Dit komt overeen met 5% van de hele verticale as. Elk tweede streepje op deze as geeft de volgende 10% aan en is iets breder getekend. Hiermee kan de lengte van de staven redelijk geschat worden.

Het programma kan maximaal 100 waarnemingen verwerken. Uit estetische overwegingen moet u echter niet veel meer dan 40 waarnemingen invoeren, omdat de staafjes anders meer op lijntjes dan op balkjes gaan lijken.

De waarnemingen worden zo 'geschaald' dat de grootste waarneming precies met 160, de lengte van de verticale as, overeenkomt.

Het zou mooi zijn als we bij de staven, de assen en de schaalverdeling tekst en getallen konden zetten, maar dat gaat heel lastig, dus doen we het nu maar niet. Denkt u erom dat de getallen die u intoetst de lengte van de staven aangeven. Als elke staaf een bepaalde klasse met waarnemingen voorstelt, dan voert u dus steeds het aantal waarnemingen (de frequentie) van een klasse in. Het programma kan uitgebreid worden door er een stuk voor te zetten dat ruwe gegevens inleest, die vervolgens netjes in klassen verdeeld worden. De frequentie van de waarnemingen in de klassen wordt vervolgens gebruikt om de stavengrafiek te tekenen.

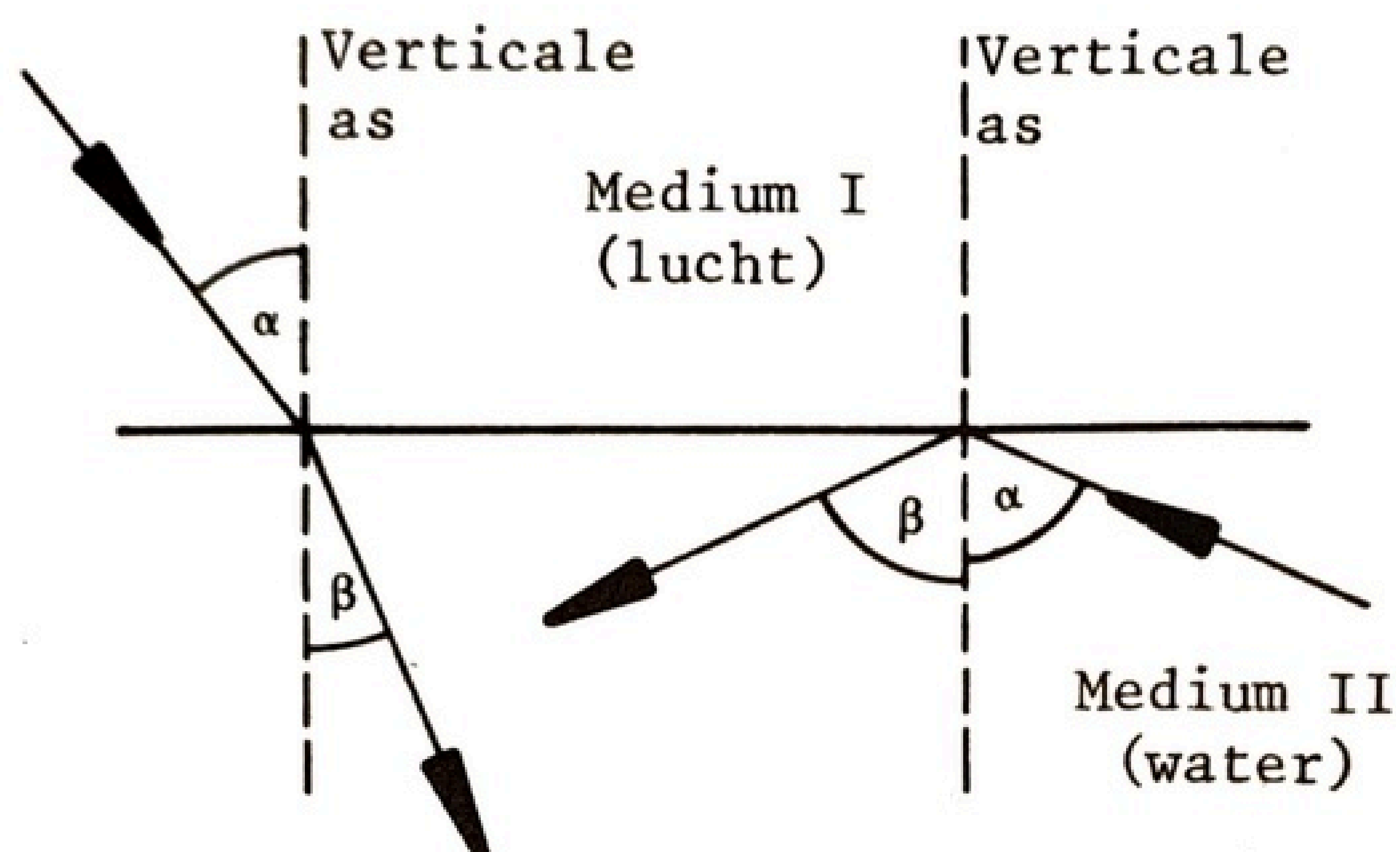


```
100 REM PROGRAMMA 37 HISTOGRAM
110 PRINT CHR$(147)
120 PRINT "HISTOGRAM TEKENEN"
130 PRINT "-----------------"
140 PRINT : PRINT
150 INPUT "HOEVEEL GEGEVENS(<40)";N
160 DIM A(N)
170 MX=-1E10 : PRINT
180 FOR J=1 TO N
190 :  PRINT "WAARDE"; J; TAB(12);
200 :  INPUT A(J)
210 :  IF A(J)>MX THEN MX=A(J)
220 NEXT J
230 PRINT CHR$(147)
240 HIRES 0,1
250 REM HORIZONTALE AS
260 LINE 0,180,320,180,1
270 REM VERTICALE AS
280 LINE 10,0,10,180,1
290 REM SCHAALVERDELING
300 FOR J=10 TO 1 STEP -1
310 :  X1=4 : Y1=180-J*16 : X2=10 :Y2=Y1
320 :  LINE X1,Y1,X2,Y2,1
330 :  X1=7 :Y1=Y2+8 : X2=10 : Y2=Y1
340 :  LINE X1,Y1,X2,Y2,1
350 NEXT J
360 REM STAVEN TEKENEN B=BREEDTE
370 B=INT(300/N) : H=0.5
380 FOR J=1 TO N
390 :  X1=(J-1)*B+15 : Y1=180
400 :  X2=X1 : Y2=INT(180-160*A(J)/MX+H)
410 :  LINE X1,Y1,X2,Y2,1
420 :  X1=X2 : Y1=Y2 : X2=X1+B : Y2=Y1
430 :  LINE X1,Y1,X2,Y2,1
440 :  X1=X2 : Y1=Y2 : X2=X1:Y2=180
450 :  LINE X1,Y1,X2,Y2,1
460 NEXT J
470 GET A$ : IF A$="" THEN 470
480 END
```

3. Demonstratieprogramma voor de breking van lichtstralen

Met dit programma willen we laten zien hoe we de Commodore 64 bij natuurkundelessen zouden kunnen gebruiken. Voordat we het programma geven leggen we nog iets uit van de natuurkundige beginselen van de breking van licht.

Als een lichtstraal vanuit de ene stof (bijvoorbeeld lucht) een andere, optisch dichtere, stof (bijvoorbeeld water) binnenkomt, wordt de straal op het scheidingsvlak van beide stoffen naar de verticale as toe gebogen. Zie onderstaande figuur.



Hierbij geldt de brekingswet van Snellius:

$$\frac{\sin\alpha}{\sin\beta} = \frac{c_1}{c_2} = n$$

c_1 en c_2 zijn de lichtsnelheden in respectievelijk de eerste en de tweede stof. De constante n heet de brekingsindex. Bij de overgang van lucht naar water geldt een brekingsindex van 1,33. Voor de overgang van lucht naar glas gelden andere waarden, enzovoorts.

Als het licht vanuit een 'dichter' medium overgaat in een 'dunner' medium geldt het omgekeerde: de lichtstralen worden nu van de verticale as, die loodrecht op het scheidingsvlak van beide stoffen staat, afgebogen. De brekingshoek α kan nooit groter dan 90° zijn.

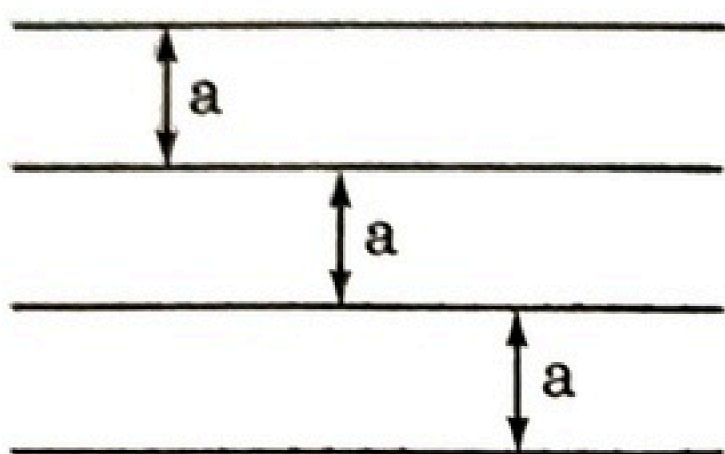
Voor dit grensgeval (α = 90°) geldt:

$$\frac{\sin 90^\circ}{\sin\beta^*} = n \Rightarrow \sin\beta^* = \frac{1}{n} \quad (\text{want } \sin 90^\circ = 1)$$

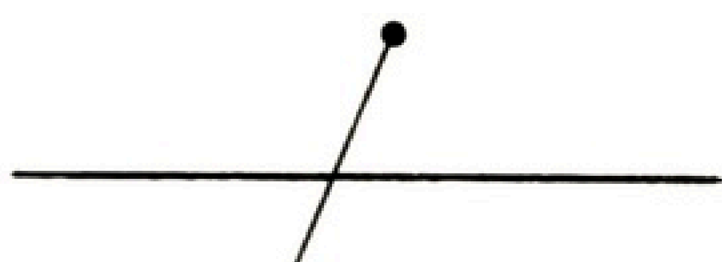
```
100 REM PROGRAMMA 38 BREKING VAN LICHT
110 PRINT CHR$(147)
120 INPUT "BREKINGSINDEX N";N
130 V=100 : H=0.5 : RD =π/180
140 HIRES 0,1
150 LINE 0,V,320,V,1
160 LINE 0,190,320,190,1
170 LINE 320,190,320,10,1
180 LINE 320,10,0,10,1
190 LINE 0,10,0,190,1
200 REM STRALEN IN MEDIUM 1 EN 2 TEKENEN
210 REM B=BETA IN GRADEN;B1=BETA IN RADIALEN
220 REM A1=ALFA IN READIALEN; SA=SIN(A1)
230 B=B+3 : B1=B*RD : X1=0 : Y1=190
240 X2=INT(90*TAN(B1)+H) : Y2=V
250 IF X2>320 THEN 400
260 LINE X1,Y1,X2,Y2,1:X1=X2:Y1=Y2
270 REM SIN(ALFA) EN ALFA BEREKENEN
280 REM CONTROLE OP TOTALE REFLECTIE
290 S=N*SIN(B1) : IF S>1 THEN 350
300 A1=ATN(S/SQR(1-S*S))
310 X2=INT(X1+90*TAN(A1)+H) : Y2=10
320 IF X2<320 THEN LINE X1,Y1,X2,Y2,1 : GOTO 230
330 X2=320 : Y2=INT(V-(320-X1)/TAN(A1)+H)
340 LINE X1,Y1,X2,Y2,1 : GOTO 230
350 REM TOTALE TERUGKAATSING
360 X2=X1+X1 : Y2=190
370 IF X2<320 THEN LINE X1,Y1,X2,Y2,1 : GOTO 230
380 X2=320 : Y2=INT(V+(320-X1)/TAN(B1)+H)
390 LINE X1,Y1,X2,Y2,1 : GOTO 230
400 GET A$ : IF A$="" THEN 400
410 END
```

4, De speldenworp van Buffon (1773)

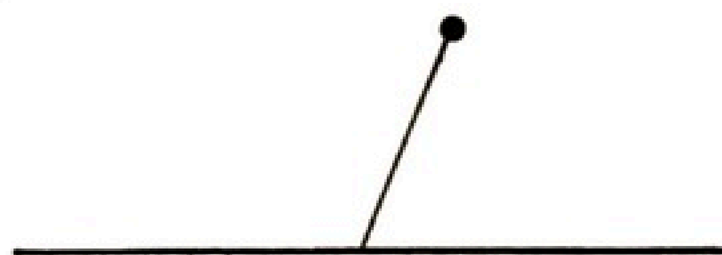
Stelt u zich voor dat in een plat vlak een aantal evenwijdige lijnen getrokken worden. Elke twee lijnen liggen op een afstand a van elkaar.



We werpen nu, zonder echt te 'mikken', een speld met een lengte c die kleiner dan of gelijk aan a is ($c \leq a$) op het vlak met de evenwijdige lijnen. Hoe groot is nu de kans dat een speld een van de lijnen treft, dat wil zeggen snijdt of raakt?



speld snijdt een lijn



speld raakt een lijn

Dit probleem kwam in 1773 op bij de Franse wiskundige Buffon na het zien van de Amerikaanse vlag met de 'stars en stripes'. Buffon heeft berekend dat deze kans gelijk is aan

$$\frac{2c}{a\pi}$$

Omdat deze formule de constante π bevat, heeft men deze 'speldenworp' vaak gebruikt om de waarde van π door simulatie te bepalen. We werpen hiertoe vaak, bijvoorbeeld een lucifer, op een papier waarop een aantal evenwijdige lijnen (met een onderlinge afstand die groter dan of gelijk aan de lengte van de lucifer is). Als de lucifer in n worpen k keer een lijn treft, dan geldt dat

$$\frac{2c}{a\pi} \quad \text{ongeveer gelijk is aan } \frac{k}{n} .$$

Beide uitdrukkingen geven immers een indruk van de kans dat de lucifer een lijn treft.
We kunnen als volgt de waarde van π benaderen:

$$\frac{2c}{a\pi} = \frac{k}{n} \Rightarrow \pi \simeq \frac{2cn}{ak} .$$

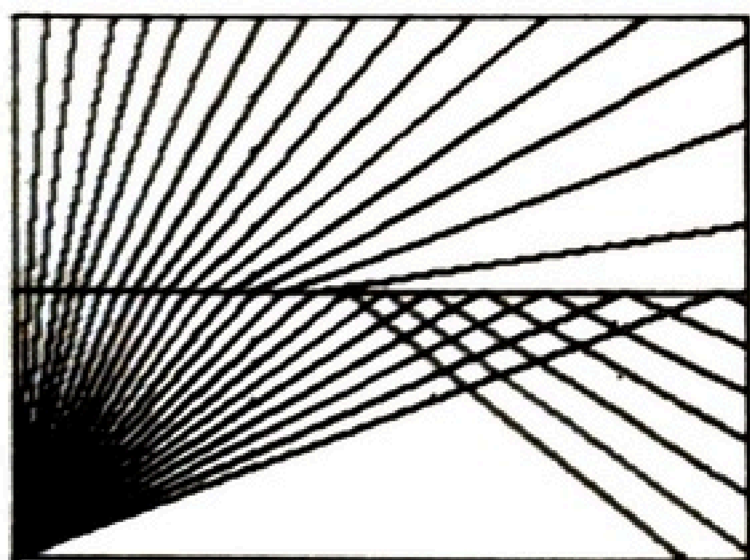
Bij de overgang van lucht naar water geldt voor β*:

$$\sin\beta^* = \frac{1}{1,33} \Rightarrow \sin\beta^* = 0,7519 \Rightarrow \beta^* = 48,75°$$

Groter dan 48,75° kan β dus niet worden. Dit betekent dat als het licht van water overgaat in lucht met een invalshoek α die groter is dan β* (zie rechter figuur), het licht niet meer het water 'uitkomt'. Op het scheidingsvlak van water en lucht wordt het licht geheel teruggekaatst. Op dit principe berusten de moderne glasvezelkabels, waarin informatie in de vorm van licht wordt getransporteerd.

In het onderstaande demonstratieprogramma kan de brekingsindex n ingetoetst worden. De lichtbron bevindt zich in het punt met schermcoördinaten (0,190). Het scheidingsvlak ligt horizontaal en is de lijn V=110. De invalshoek van een lichtstraal die van medium 2 (de dichtere stof, onderste helft) in medium 1 (de lichtere stof, bovenste helft) overgaat wordt van 0° steeds met stapjes van 3° opgehoogd. Op het moment dat deze invalshoek groter wordt dan β* (deze is afhankelijk van de ingetoetste brekingsindex) treedt totale reflectie (terugkaatsing) op. Rond de figuur wordt een kader getekend. De eindpunten van de diverse lichtstralen worden met trigonometrische functies berekend. De commentaaropdrachten in het programma leggen nog eens uit 'wat waar' gebeurt.

De inverse functie van de sinusfunctie (de boogsinus of arcsinus) bestaat niet in Microsoft BASIC. We lossen dit op door de inverse tangensfunctie (ATN in BASIC) te gebruiken.

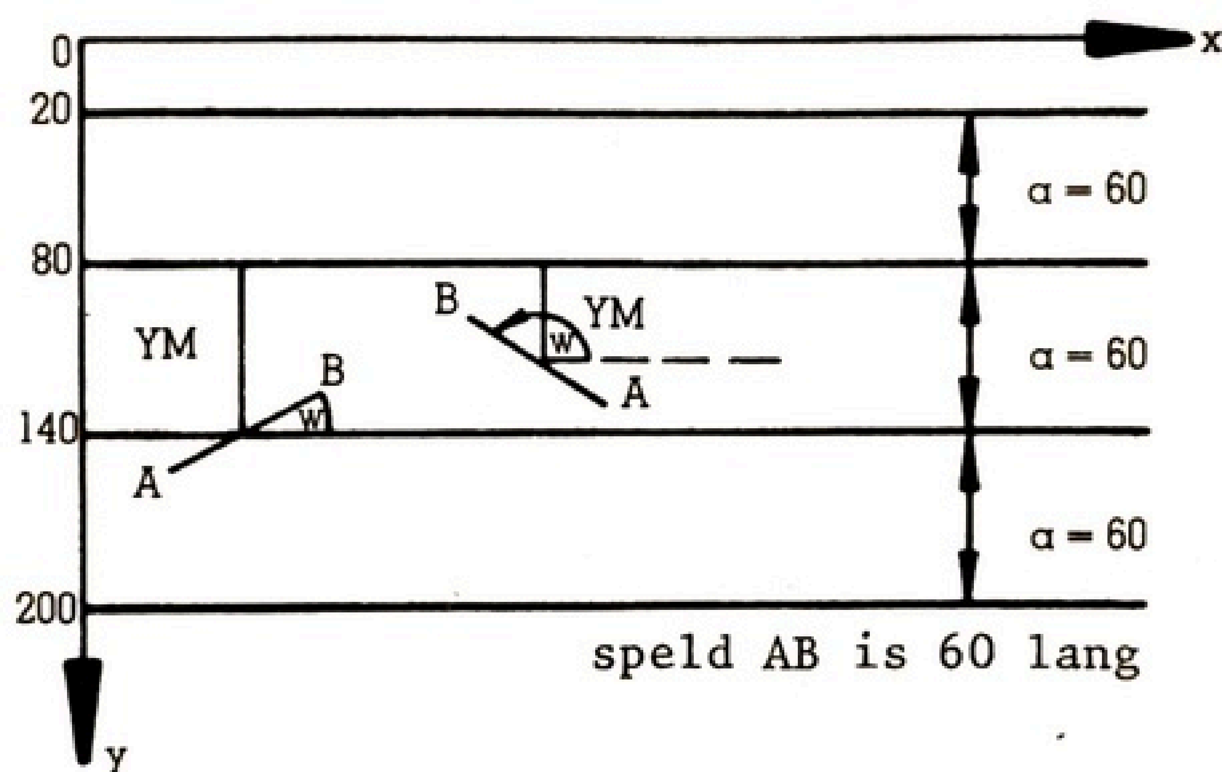


Het voordeel van een dergelijke computersimulatie, boven het uitvoeren van het natuurkundige experiment, is dat we heel gemakkelijk verschillende brekingsindexen kunnen invoeren zonder steeds andere apparatuur op te hoeven stellen en dat de lichtstralen als dunne lijnen zichtbaar gemaakt kunnen worden.

Op deze manier heeft de astronoom Rudolf Wolf in 1850 met 5000 luciferworpen voor π de waarde 3,1596 gevonden (π = 3,141592654....).

Met een computer kunnen we gemakkelijk duizenden worpen simuleren. Deze programma's vinden we in bijna elk informaticaleerboek. Dit zijn echter programma's die vrij lang draaien, maar waaraan niets te 'zien' is. Als we bijvoorbeeld in zo'n programma voor n de waarde 10.000 zouden intikken, zien we eerst een tijd niets en vervolgens verschijnt de mededeling dat van de 10.000 keer 6.349 keer een lijn is geraakt, hetgeen voor π de waarde 3,1501024 oplevert.

We gaan een programma maken dat ook dergelijke berekeningen uitvoert, maar dat bovendien elke speld die geworpen wordt laat zien. We zien het experiment als een film aan ons voorbijgaan. Bekijk hiertoe de onderstaande tekening.



De manier waarop een speld valt kan met twee toevalsgetallen bepaald worden. De twee toevalsgetallen bepalen respectievelijk de afstand YM van de speld tot de daarboven liggende lijn en de hoek W die de speld met de lijnen maakt:

$0 \leq YM \leq a$	;	YM is de afstand van het midden van de speld tot de daarboven liggende horizontale lijn
$0 \leq W \leq 180°$	;	W is de hoek (in positieve zin, dat wil zeggen bij draaiing tegen de klok in) die de speld met de positieve x-richting maakt

Hierna volgt een illustratie van hoe het eruit kan zien en het programma.



```
100 REM PROGRAMMA 39 SPELDENWORP VAN BUFFON
110 PRINT CHR$(147)
120 PRINT "SPELDENWORP VAN BUFFON"
130 PRINT "----------------------"
140 INPUT "HOEVEEL WORPEN"; N
150 M=0 : H=0.5
160 PRINT CHR$(147)
170 HIRES 0,1
180 REM LIJNEN TEKENEN
190 FOR Y1=10 TO 190 STEP 60
200 :  LINE 0,Y1,320,Y1,1
210 NEXT Y1
220 REM N MAAL GOOIEN EN TEKENEN
230 FOR J=1 TO N
240 :  XM=INT(290*RND(1)+30+H)
250 :  YM=INT(60*RND(1)+70+H)
260 :  W=*RND(1)
270 :  DX=30*COS(W) : DY=30*SIN(W)
280 :  X1=INT(XM-DX+H):Y1=INT(YM+DY+H)
290 :  X2=INT(XM+DX+H):Y2=INT(YM-DY+H)
300 :  LINE X1,Y1,X2,Y2,1
310 :  IF Y1>=140 OR Y2<=80 THEN M=M+1
320 NEXT J
330 GET A$ : IF A$="" THEN 330
340 PRINT CHR$(147)
350 PRINT "AANTAL WORPEN       "; N
360 PRINT "AANTAL KEER SNIJDEN "; M
370 PRINT "BENADERING VOOR     "; 2*N/M
380 END
```

Uit de waarden voor YM en W berekenen we de coördinaten van de uiteinden A en B van de speld. Het programma tekent hiermee de speld op het beeldscherm.

Een speld treft een lijn als de y-coördinaat van A groter dan gelijk aan 140 is of als de y-coördinaat van B kleiner dan of gelijk aan 80 is. De x-coördinaten van A en B doen er in het geheel niet toe.

Het programma tekent als 'speelveld' vier evenwijdige lijnen met een onderlinge afstand van 60. De spelden zijn trouwens ook 60 lang. Als alle spelden geworpen zijn kunnen door een toets in te drukken de waarden voor n, k en π worden afgelezen.

5. Prooi-roofdierpopulaties

Het laatste educatieve programma is een (deterministische) simulatie van een ecologisch systeem. Het is een demonstratieprogramma voor een biologieles.

Het ecologische systeem bevat gras, hazen en vossen. Tussen deze drie ecologische componenten gelden de volgende betrekkingen:

1. De hazen eten gras en de vossen eten hazen.
2. Als er meer gras groeit, neemt ook het aantal hazen toe. Deze hazen eten echter van het gras en verminderen zo hun eigen groei.
3. Als er meer hazen komen, neemt ook het aantal vossen toe. Omdat vossen hazen eten, verminderen zij zelf hun groei, net zoals bij de hazen en het gras.

Als we het aantal hazen op een bepaald tijdstip t aangeven met h(t) en het aantal vossen met v(t), kunnen we, volgens Lotka en Volterra (1920), voor het aantal hazen en vossen op tijdstip t+1 de volgende vergelijkingen opstellen:

$$h(t+1) = h(t) + a.h(t) - b.h(t).v(t) \qquad \text{vergelijking (1)}$$
$$v(t+1) = v(t) + c.v(t).h(t) - d*v(t) \qquad \text{vergelijking (2)}$$

De toename van het aantal hazen tussen de tijdstippen t en t+1 is evenredig met het aantal hazen op tijdstip t, dus

$$\underbrace{h(t+1)-h(t)}_{\text{toename hazen}} = a.h(t)$$

a: groeifactor
h(t): aantal hazen op tijdstip t

De afname van het aantal hazen is evenredig met het aantal aanwezige hazen (natuurlijk verloop) en met het aantal vossen, want die eten hazen, dus

$$\underbrace{h(t+1)-h(t)}_{\text{groei van het aantal hazen}} = \underbrace{a.h(t)}_{\text{toename aantal hazen}} - \underbrace{b.h(t).v(t)}_{\text{afname aantal hazen}}$$

Dit is vergelijking (1). We nemen in het programma aan dat er altijd genoeg gras voor de hazen voorhanden is.

De toename van het aantal vossen is evenredig met het aantal aanwezige vossen en met het aantal hazen (prooi). De afname van het aantal vossen is alleen evenredig met het aantal, omdat in dit systeem de vossen zelf geen prooidieren zijn. Voor de vossen krijgen we dus:

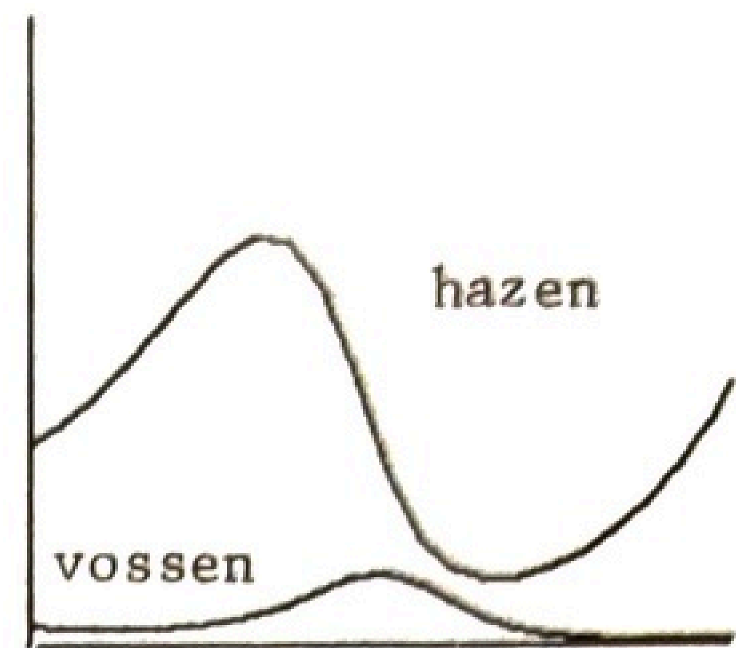
$$\underbrace{v(t+1)-v(t)}_{\text{groei van het aantal vossen}} = \underbrace{c.v(t).h(t)}_{\text{toename aantal vossen}} - \underbrace{d.v(t)}_{\text{afname aantal vossen}}$$

De vergelijkingen zijn als zogeheten differentievergelijkingen opgesteld. Het gras speelt, zoals gezegd, eigenlijk geen rol; er is altijd genoeg om alle hazen te voeden.

Bij het draaien van het simulatieprogramma hebben we de volgende waarden gekozen:
X (= h(0)) = 200; Y (= v(0)) = 20; a=0,3; b=0,01; c=0,002 en d=0,5.

De grafiek die het programma tekent laat heel mooi de groei en de afname van de hazen- en vossenpopulaties zien. Het aantal hazen groeit eerst, bereikt een maximum en neemt vervolgens af. De vossen groeien ook, maar later, bereiken later een maximum en nemen dan ook af, waarna de cyclus zich herhaalt. In de biologie noemen we dit een dynamisch evenwicht.

U hoeft de waarden voor de coëfficiënten a, b, c en d maar iets te veranderen of het systeem kan ontregeld worden. Hierbij neemt òf het aantal hazen enorm toe òf alle hazen en vossen sterven snel uit. Deze eenvoudige simulatie toont aan hoe desastreus een kleine ingreep in een bestaand ecologisch systeem dat in een dynamisch evenwicht is kan zijn.




```
100 REM PROGRAMMA 40 ROOFDIER-PROOIDIER-SYSTEEM
110 PRINT CHR$(147)
120 INPUT "BEGINPOPULATIE PROOIDIEREN (200)"; X
130 INPUT "BEGINPOPULATIE ROOFDIEREN (20)"; Y
140 INPUT "GROEIFACTOR PROOIDIEREN (.3)"; A
150 INPUT "AFNAMEFACTOR PROOIDIEREN (.01)"; B
160 INPUT "GROEIFACTOR ROOFDIEREN (.002)"; C
170 INPUT "AFNAMEFACTOR ROOFDIEREN (.5)"; D
180 PRINT CHR$(147)
190 HIRES 0,1
200 K=0.3 : H=0.5 : V=200
210 FOR X1=0 TO 320 STEP 10
220 :   XP=X+(A*X-B*X*Y):YR=Y+(C*X*Y-D*Y)
230 :   REM POP.PROOIDIEREN TEKENEN
240 :   Y1=INT(V-K*X+H)
250 :   X2=X1+10 : Y2=INT(V-K*XP+H)
260 :   IF Y2<0 OR Y2>200 THEN 350
270 :   LINE X1,Y1,X2,Y2,1
280 :   REM POP.ROOFDIEREN TEKENEN
290 :   Y1=INT(V-K*Y+H)
300 :   X2=X1+10 : Y2=INT(V-K*YR+H)
310 :   IF Y2<0 OR Y2>200 THEN 350
320 :   LINE X1,Y1,X2,Y2,1
330 :   X=XP : Y=YR
340 NEXT X1
350 GET A$ : IF A$="" THEN 350
360 END
```

Appendix 1

BASIC-routines als alternatief voor HIRES en LINE

De opdrachten HIRES 0,1 en LINE X1,Y1,X2,Y2,1 zijn opdrachten uit Simon's BASIC. Beschikt u niet over deze BASIC-versie voor uw Commodore, dan zult u op een bepaalde manier de HIgh RESolution stand (HIRES) moeten inschakelen en ook zult u twee punten (X1,Y1) en (X2,Y2) door een rechte LINE met elkaar moeten verbinden. Dit kan op twee manieren, waarvan we de eerste in deze appendix laten zien. Deze twee manieren zijn:

1. Schrijf HIRES en LINE als een reeks 'gewone' BASIC-opdrachten.

2. Schrijf HIRES en LINE als twee machinetaalprogramma's.

De tweede mogelijkheid, die veruit te verkiezen is boven de eerste, komt in Appendix 2 aan de orde. Het nadeel, als we HIgh RESolution opdrachten in gewoon BASIC programmeren, is dat de snelheid waarmee getekend wordt zeer laag is vergeleken met de 'voorgeprogrammeerde' HIRES- en LINE-opdrachten uit Simon's BASIC en vergeleken met de in Appendix 2 gegeven machinetaalroutines (die u met een BASIC-programma kunt inlezen!). Mocht u over een BASIC (DTL)-compiler beschikken, dan kunt u de in deze appendix gegeven BASIC-programma's voor HIRES en LINE nog een factor 5 tot 10 versnellen, maar het blijft een erg langzame oplossing. Wij denken dat vrijwel alle lezers die niet over Simon's BASIC beschikken de machinetaalroutines uit Appendix 2 zullen gebruiken. Toch is het wellicht leuk te zien hoe het kiezen van de Hoge-Resolutiestand en het trekken van een lijn tussen twee punten er in gewoon BASIC uitzien!

We geven hierna het eerste programma uit dit boek, waarin we de HIRES 0,1 en LINE X1,Y1,X2,Y2,1 opdrachten vervangen hebben door een aantal subroutines in BASIC.

```
100 REM DIAGONAALWEB
110 PRINT CHR$(147)
120 GOSUB 2000:REM IN PLAATS VAN HIRES
130 Y1=0 : Y2=200
200 FOR X1=0 TO 320 STEP 40
210 :  FOR X2=0 TO 320 STEP 40
220 :    GOSUB 3000:REM I.P.V. LINE
230 :  NEXT X2
240 NEXT X1
300 GET A$ : IF A$="" THEN 300
310 END
2000 POKE 53272,PEEK(53272)OR8
2010 FOR VA=8192 TO 16191
2020 :  POKE VA,0
2030 NEXT VA
2040 FOR VA=1024 TO 2023
2050 :  POKE VA,1
2060 NEXT VA
2070 POKE 53265,PEEK(53265)OR32
2080 RETURN
3000 IF ABS(X2-X1)>ABS(Y2-Y1) THEN GOSUB 4000:RETURN
3010 GOSUB 5000
3020 RETURN
4000 FOR PX=X1 TO X2 STEP SGN(X2-X1)
4010 :  PY=(Y2-Y1)/(X2-X1)*(PX-X1) + Y1
4020 :  GOSUB 6000
4030 NEXT PX
4040 RETURN
5000 FOR PY=Y1 TO Y2 STEP SGN(Y2-Y1)
5010 :  PX=(X2-X1)/(Y2-Y1)*(PY-Y1) + X1
5020 :  GOSUB 6000
5030 NEXT PY
5040 RETURN
6000 VY=320*INT(PY/8)+(PYAND7)
6010 VX=8*INT(PX/8)
6020 VA=8192 + VX+VY
6030 MA=2↑(7-(PXAND7))
6035 POKE VA,PEEK(VA) OR MA
6040 RETURN
```

We zien het oorspronkelijke programma (regels 100-310) waarin in regel 120:

HIRES 0,1 vervangen is door GOSUB 2000

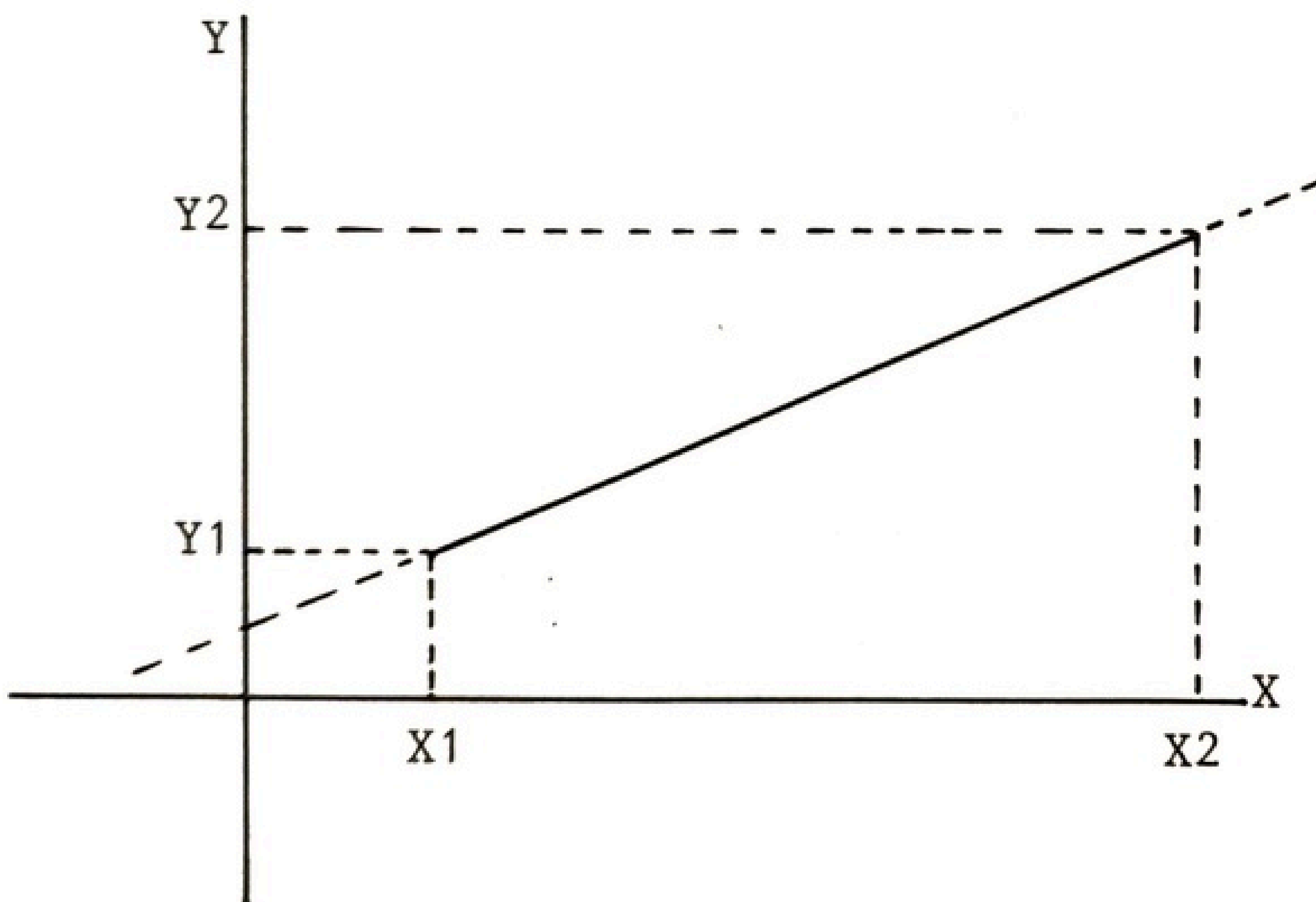
en in regel 220:

LINE X1,Y1,X2,Y2,1 vervangen is door GOSUB 3000

Subroutine 2000 (regels 2000-2080) zorgen ervoor dat de Commodore in de zogeheten BIT-MAP mode komt en dat het graphics-scherm wordt schoongemaakt en dat het scherm wit gemaakt wordt en dat de afdrukkleur zwart wordt. Dit is precies wat HIRES 0,1 ook doet.

De subroutines 3000, 4000, 5000 en 6000 tekenen een 'zwarte' lijn tussen de punten (X1,Y1) en (X2,Y2), het alternatief voor LINE X1,Y1,X2,Y2,1.

Hieronder zien we in het xy-vlak een tekening van een lijn(stuk) tussen (X1,Y1) en (X2,Y2).



De vergelijking van de lijn door (X1,Y1) en (X2,Y2) is

$$Y-Y1 = \frac{Y2-Y1}{X2-X1} \cdot (X-X1)$$

We tekenen deze lijn op het (320x200)-puntjesscherm door tussen X1 en X2 (of tussen (Y1 en Y2) de X-waarde (of de Y-waarde)

steeds met 1 (één beeldschermpunt) te laten oplopen en de daarbij horende Y-waarde (of X-waarde) te berekenen.

Als de afstand van X1 tot X2 groter is dan de afstand tussen Y1 en Y2, laten we X oplopen (regel 4000) en berekenen de daarbij behorende Y-waarde (regel 4010). Als de afstand tussen Y1 en Y2 groter dan of gelijk aan de afstand tussen X1 en X2 is, laten we Y oplopen (regel 5000) en berekenen de daarbij horende X-waarde (regel 5010). Deze test wordt uitgevoerd in regel 3000, waarin de absolute waarden van X2-X1 en Y2-Y1 (de afstanden tussen X1 en X2 en tussen Y1 en Y2) worden vergeleken.

Als de beeldschermcoördinaten PX en PY van een punt berekend zijn, wordt dit punt getekend door op de juiste plaats in het hoge-resolutie-videogeheugen de juiste waarde te poken (regel 6035). Meer hierover vindt u in de *Programmers reference guide* van uw Commodore 64 (hoofdstuk 3, pp.121-127).

Mocht u deze routines 2000 t/m 6000 bij elk programma willen gebruiken, maak er dan een programma van dat u op cassetteband (of op diskette) opslaat. Lees dan steeds voor u een programma uit dit boek gaat intoetsen eerst deze routines van cassette of van diskette in en toets daarna het programma uit het boek in. Vervang hierin HIRES 0,1 door GOSUB 2000 en LINE X1,Y1,X2,Y2,1 door GOSUB 3000. Zorg dat de coördinaten van de twee uiteinden van het te tekenen lijnstuk altijd door X1,Y1 en X2,Y2 weergegeven worden.

Mocht u niet met Simon's BASIC maar met Supergraphik 64 werken, dan moet u

HIRES 0,1 vervangen door GMODE 0,1:GCLEAR

en

LINE X1,Y1,X2,Y2,1 vervangen door PLOT X1,Y1 TO X2,Y2

Hierna geven we nog een iets snellere versie waarin we de subroutines 3000 t/m 6000 'samengeperst' hebben tot één subroutine 3000. Dit geeft slechts een kleine verbetering in de snelheid.

In de volgende appendix geven we een BASIC-programma dat acht(!) Hoge-Resolutie-routines als machinetaalroutines in het geheugen zet. Hiermee kunt u zeer snel tekenen; even snel als bijvoorbeeld in Simon's BASIC.

```
100 REM DIAGONAALWEB
110 PRINT CHR$(147)
120 GOSUB 2000:REM IN PLAATS VAN HIRES
130 Y1=0 : Y2=200
200 FOR X1=0 TO 320 STEP 40
210 :  FOR X2=0 TO 320 STEP 40
220 :     GOSUB 3000:REM I.P.V. LINE
230 :  NEXT X2
240 NEXT X1
300 GET A$ : IF A$="" THEN 300
310 END
2000 POKE 53272,PEEK(53272)OR8
2010 FOR VA=8192 TO 16191
2020 :  POKE VA,0
2030 NEXT VA
2040 FOR VA=1024 TO 2023
2050 :  POKE VA,1
2060 NEXT VA
2070 POKE 53265,PEEK(53265)OR32
2080 RETURN
3000 DX=X2-X1:DY=Y2-Y1:SX=SGN(DX):SY=SGN(DY)
3010 IF ABS(DX)<=ABS(DY) THEN GOTO 3080
3020 FOR PX=X1 TO X2 STEP SX:PY=DY/DX*(PX-X1)+Y1
3031 VA=8192+8*INT(PX/8)+320*INT(PY/8)+(PYAND7)
3032 MA=2↑(7-(PXAND7)):POKE VA,PEEK(VA)ORMA
3050 NEXT PX:RETURN
3080 FOR PY=Y1 TO Y2 STEP SY:PX=DX/DY*(PY-Y1)+X1
3101 VA=8192+8*INT(PX/8)+320*INT(PY/8)+(PYAND7)
3102 MA=2↑(7-(PXAND7)):POKE VA,PEEK(VA)ORMA
3110 NEXT PY:RETURN
```

Appendix 2

BASIC-programma voor het inlezen van acht zeer snelle hoge-resolutie-routines in machinecode

Het nu volgende, zeer lange, programma, dat oorspronkelijk verschenen is in het blad Commodore Computing (oktober 1983), is het ei van Columbus voor diegenen die alleen over de standaard Commodore 64 BASIC beschikken. In het 65.536 (64K) geheugenplaatsen tellende geheugen van de Commodore 64 is niet alleen ruimte voor het opslaan van BASIC-programma's maar ook voor machinetaalprogramma's. Brengen we een (snel) machinetaalprogramma in het geheugen, dan kunnen we deze routine vanuit een BASIC-programma laten uitvoeren. In deze appendix zullen we laten zien hoe dit allemaal gerealiseerd kan worden, zelfs als we niets van machinecode afweten!

Een machinetaalprogramma is een programma waarin alle opdrachten uit groepjes van 8 nullen en énen (bits) bestaan. Dat geen mens in nullen en énen programmeert zal niemand verbazen. Een eerste verbetering is zo'n binaire code van 8 bits te schrijven als een hexadecimale of decimale code. We kunnen dan een programma schrijven als een reeks decimale getallen. Elk getal is een opdrachtcode voor de computer, die weet wat hem bij elke code te doen staat.

Als we een 8-bit-code als decimaal getal schrijven, krijgen we een getal tussen 0 (code 00000000) en 255 (code 11111111). Zo'n code past precies in één geheugenplaats. Het in het geheugen zetten van een machinetaalprogramma is niets anders dan het vullen van een aantal opéénvolgende geheugenplaatsen met de (decimale, binaire of hexadecimale) codes van het machinetaalprogramma.

BASIC kent de POKE-opdracht waarmee we een code (tussen 0-255) in een bepaalde geheugenplaats kunnen zetten. Als we de decimale codes uit een machinetaalprogramma in DATA-regels opnemen en we gebruiken in een FOR-NEXT-opdracht de READ-opdracht om steeds een code in te lezen en in een geheugenplaats te POKEn, kopiëren

we als het ware het machinetaalprogramma in een bepaald stukje geheugen.

In de Commodore 64 is vanaf geheugenplaats 49152 (na 48K) ruimte gereserveerd voor het opslaan van machinetaalprogramma's. Het nu volgende BASIC-programma bevat een achttal high-resolution-routines in machinecode die als één sliert van 2059 decimale codes in DATA-regels zijn opgenomen. De eerste code uit het programma (code 32 in DATA-regel 49152) komt terecht op de geheugenplaats met het adres 49152; de tweede code (253) komt op adres 49153; de derde (174) op adres 49154, enzovoorts. Het machinetaalprogramma is verdeeld in regels met steeds zeven codes. Alleen de laatste machinetaalcode (199 in DATA-regel 51210) staat in zijn eentje.

Om het intoetsen van deze 2059 codes enigszins in goede banen te leiden is om de 10 DATA-regels een CONTROLE-regel opgenomen. Zo'n controleregel, bijvoorbeeld DATA-regel 49216, bevat een getal en een string. Het getal (8560 in regel 49216) is steeds de som van de 70 machinecodes in de daarvoor staande 10 DATA-regels. De laatste controleregel 51211 bevat de som van de laatste 41 machinecodes.

Als het programma ingetoetst is en draait, worden steeds de ingelezen en gePOKEte (regels 170 en 200) machinecodes bij elkaar opgeteld (regel 210). Dit gaat door tot het einde van de invoer gelezen is (regel 180), hetgeen alleen aan het einde van de DATA-regels (51211) gebeurt of tot een code groter dan 255 gelezen wordt (regel 190). Zo'n code >255 is altijd het controlegetal uit zo'n controleregel. Wordt zo'n controlegetal gelezen dan vergelijkt de computer (regel 250) of dit controlegetal gelijk is aan de som van de daarvoor staande machinecodes. Is dit zo, dan zitten er geen fouten in de desbetreffende DATA-regels. Zo niet, dan zit er ergens in de maximaal 10 voorafgaande DATA-regels een foute code.

Het is de bedoeling dat u het onderstaande programma één keer foutloos intikt en het daarna op cassetteband of op diskette opslaat. U kunt hierbij het beste als volgt tewerkgaan: allereerst tikt u de regels 100 t/m 310 in, gevolgd door de eerste DATA-regel 49152. Als laatste getal in deze DATA-regel voegt u het getal -1 toe. U draait het programma en kijkt of het met die ene DATA-regel goed werkt. Vervolgens verwijdert u de -1 uit DATA-regel 49152 en toetst de regels 49159-49216 in. Regel 49216 sluit u af met -1, dus

```
49216  DATA 8560,"REGELS 49152-49215",-1
```

U draait het programma opnieuw. Als alle codes goed zijn ingetikt ziet u op het scherm de melding

```
8560  8560  REGELS 49152-49215
```

Hebt u bijvoorbeeld in DATA-regel 49215 de code 273 getikt in plaats van 173, dan krijgt u de melding

```
8660  8560  REGELS 49152-49215
FOUT IN REGELS 49152-49215
FOUT(EN) GECONSTATEERD
READY.
```

Hieraan kunt u zien dat de som van de ingelezen codes 100 meer is (8660) dan hij moet zijn (8560). U moet dus op zoek naar de code die u fout getikt hebt. Nooit hoeft u hiervoor meer dan 10 DATA-regels te doorzoeken.

U verbetert en draait het programma opnieuw. Als u de melding GEEN FOUT ziet, zijn alle tot dan toe ingetikte codes vrijwel zeker goed. Theoretisch kunt u bijvoorbeeld ergens in plaats van 174 de code 184 intikken en in de volgende regel 131 in plaats van 141 Hierdoor blijft de som gelijk maar zijn wel twee codes verkeerd ingetoetst.

Goed, u hebt de regels t/m 49216 goed ingetoetst. U haalt nu de -1 uit regel 49216 weg (belangrijk!!) en kopieert het programma op cassette of diskette. U weet dan zeker dat het niet meer verloren kan gaan.

Vervolgens tikt u de regels 49222 t/m 49286 in, sluit regel 49286 af met -1, draait het programma en verbetert de eventueel in de regels 49222 t/m 49286 gemaakte fouten. U verwijdert de -1 en maakt een nieuwe kopie van het programma. Zo gaat u door tot en met regel 51211. Bij deze werkwijze kunnen hoogstens 10 DATA-regels verloren gaan. Tik nooit alles in één keer in zonder tussentijds het programma op cassette of diskette te kopiëren.

Als u alles goed hebt ingetikt, krijgt u bij het draaien van het programma de op p.117 afgebeelde lijst op het scherm te zien.

Wij hebben de DATA-regels genummerd van 49152 t/m 51211, waarbij we steeds de nummers met 7 opgehoogd hebben. Op deze manier komen de nummers van de DATA-regels precies overeen met het adres van de geheugenplaats waar de eerste code uit een DATA-regel terechtkomt. Zo komt bijvoorbeeld de code 240 uit DATA-regel 49257 op geheugenadres 49257; de code 5 op adres 49258, code 198 op adres 49259, 87 op adres 49260, 76 op 49261, 99 op 49262 en 192 op adres 49263.

```
8560        8560        REGELS 49152-49215
8560        8560        REGELS 49222-49285
9637        9637        REGELS 49292-49355
7927        7927        REGELS 49362-49425
7568        7568        REGELS 49432-49495
7438        7438        REGELS 49502-49565
9049        9049        REGELS 49572-49635
7846        7846        REGELS 49642-40705
7521        7521        REGELS 40712-49775
7086        7086        REGELS 49782-49845
7605        7605        REGELS 49852-49915
6680        6680        REGELS 49922-49985
6728        6728        REGELS 49992-50055
6013        6013        REGELS 50062-50125
6293        6293        REGELS 50132-50195
5994        5994        REGELS 50202-50265
5826        5826        REGELS 50272-50335
5995        5995        REGELS 50342-50405
5432        5432        REGELS 50412-50475
9018        9018        REGELS 50482-50545
8397        8397        REGELS 50552-50615
8696        8696        REGELS 50622-50685
7501        7501        REGELS 50692-50755
7123        7123        REGELS 50762-50825
8822        8822        REGELS 50832-50895
7824        7824        REGELS 50902-50965
7727        7727        REGELS 50972-51035
7733        7733        REGELS 51042-51105
7102        7102        REGELS 51112-51175
3054        3054        REGELS 51182-51210
GEEN FOUT
```

Zo kunt u altijd nagaan welke code op welk adres terechtkomt. Om dit consequent vol te houden zijn de controleregels genummerd met één hoger dan de daarvoor staande regel. Deze manier van regelnummering kan eenvoudig gerealiseerd worden als u beschikt over de AUTO-opdracht voor het automatisch nummeren van programmaregels. U kunt echter ook uw eigen regelnummering kiezen voor de DATA-regels, bijvoorbeeld vanaf 1000, of 20C00, of ..... . U moet dan wel de strings in de controleregels overeenkomstig aanpassen.

Nu volgt het programma. Om het intikken te vergemakkelijken hebben we na elke controleregel een lege regel afgedrukt.

```
100 REM PROGRAMMA VOOR HET POKEN VAN ACHT
110 REM HIGH-RESOLUTION MACHINETAAL PROGRAMMA'S.
120 REM DE MACHINE CODE BEVINDT ZICH IN DE DATAREGELS
130 REM 49152 T/M 51210. DEZE REGELNUMMERS KOMEN
140 REM OVEREEN MET HET ADRES VAN DE GEHEUGENPLAATS
150 REM WAAR DE EERSTE CODE UIT DE DATAREGEL GEPOKED WORDT
160 I=49152 : T=0 : PRINT CHR$(147)
170 READ A
180 :    IF A=-1 THEN 290
190 :    IF A>255 THEN 230
200 :    POKE I,A
210 :    T=T+A : I=I+1 : GOTO 280
220 :    REM TUSSENCONTROLE
230 :       READ A$
240 :       PRINT T,A,A$
250 :       IF T=A THEN T=0 : GOTO 280
260 :       ER=1 : T=0
270 :       PRINT "FOUT IN "; A$
280 GOTO 170
290 IF ER=0 THEN PRINT "GEEN FOUT"
300 IF ER=1 THEN PRINT "FOUT(EN) GECONSTATEERD"
310 END
49152 DATA 32,253,174,32,235,183,138
49159 DATA 141,32,208,141,33,208,165
49166 DATA 20,133,251,240,2,162,0
49173 DATA 32,89,192,32,128,192,32
49180 DATA 166,192,169,59,141,17,208
49187 DATA 169,29,141,24,208,165,251
49194 DATA 240,5,169,216,141,22,208
49201 DATA 169,128,133,56,133,52,173
49208 DATA 2,221,9,3,141,2,221
49215 DATA 173,0,221,41,252,9,1
49216 DATA 8560,"REGELS 49152-49215"

49222 DATA 141,0,221,169,132,141,136
49229 DATA 2,169,79,141,17,3,169
49236 DATA 197,141,18,3,96,160,0
49243 DATA 169,64,133,87,169,191,133
49250 DATA 88,169,0,145,87,165,87
49257 DATA 240,5,198,87,76,99,192
49264 DATA 198,88,165,88,201,159,240
49271 DATA 7,169,255,133,87,76,99
49278 DATA 192,96,160,0,169,231,133
49285 DATA 87,169,135,133,88,138,145
49286 DATA 8560,"REGELS 49222-49285"
```

```
49292 DATA 87,165,87,240,5,198,87
49299 DATA 76,138,192,198,88,165,88
49306 DATA 201,131,240,7,169,255,133
49313 DATA 87,76,138,192,96,160,0
49320 DATA 169,231,133,87,169,219,133
49327 DATA 88,169,0,145,87,165,87
49334 DATA 240,5,198,87,76,176,192
49341 DATA 198,88,165,88,201,215,240
49348 DATA 7,169,255,133,87,76,176
49355 DATA 192,96,165,90,201,0,240
49356 DATA 9637,"REGELS 49292-49355"

49362 DATA 16,201,1,208,6,165,89
49369 DATA 201,64,144,6,32,181,194
49376 DATA 76,72,178,165,91,201,200
49383 DATA 144,6,32,181,194,76,72
49390 DATA 178,165,89,41,7,133,94
49397 DATA 169,7,56,229,94,133,94
49404 DATA 165,251,240,11,70,94,6
49411 DATA 94,165,94,24,105,1,133
49418 DATA 95,165,94,240,9,168,169
49425 DATA 1,10,136,208,252,240,2
49426 DATA 7927,"REGELS 49362-49425"

49432 DATA 169,1,133,94,165,95,240
49439 DATA 9,168,169,1,10,136,208
49446 DATA 252,240,2,169,1,133,95
49453 DATA 169,0,133,92,133,88,133
49460 DATA 87,165,91,41,7,133,93
49467 DATA 165,91,74,74,74,133,91
49474 DATA 160,5,24,10,38,88,136
49481 DATA 208,249,133,87,165,91,160
49488 DATA 3,24,10,136,208,251,133
49495 DATA 91,24,101,87,133,91,165
49496 DATA 7568,"REGELS 49432-49495"

49502 DATA 88,105,0,133,92,160,3
49509 DATA 24,70,90,102,89,136,208
49516 DATA 248,165,89,133,87,165,90
49523 DATA 133,88,160,3,24,6,87
49530 DATA 38,88,136,208,248,160,8
49537 DATA 24,165,91,101,87,133,87
49544 DATA 165,92,101,88,133,88,136
49551 DATA 208,240,24,165,93,101,87
49558 DATA 133,87,169,0,101,88,24
49565 DATA 169,160,101,88,133,88,24
49566 DATA 7438,"REGELS 49502-49565"
```

```
49572 DATA 165,89,101,91,133,91,169
49579 DATA 0,101,92,133,92,96,32
49586 DATA 253,174,32,235,183,165,20
49593 DATA 133,89,165,21,133,90,138
49600 DATA 133,91,32,253,174,32,235
49607 DATA 183,138,133,252,165,20,133
49614 DATA 253,32,205,192,169,54,133
49621 DATA 1,165,251,208,3,76,123
49628 DATA 194,165,252,201,0,240,17
49635 DATA 201,1,240,40,201,2,240
49636 DATA 9049,"REGELS 49572-49635"

49642 DATA 81,201,3,240,114,169,55
49649 DATA 133,1,96,160,0,165,94
49656 DATA 73,255,133,94,165,95,73
49663 DATA 255,133,95,177,87,37,94
49670 DATA 37,95,145,87,169,55,133
49677 DATA 1,96,160,0,165,95,73
49684 DATA 255,133,95,177,87,5,94
49691 DATA 37,95,145,87,169,132,24
49698 DATA 101,92,133,92,6,253,6
49705 DATA 253,6,253,6,253,177,91
49706 DATA 7846,"REGELS 49642-40705"

49712 DATA 41,15,24,101,253,145,91
49719 DATA 169,55,133,1,96,160,0
49726 DATA 165,94,73,255,133,94,177
49733 DATA 87,37,94,5,95,145,87
49740 DATA 24,169,132,101,92,133,92
49747 DATA 177,91,41,240,24,101,253
49754 DATA 145,91,169,55,133,1,96
49761 DATA 160,0,177,87,5,94,5
49768 DATA 95,145,87,169,216,24,101
49775 DATA 92,133,92,165,253,145,91
49776 DATA 7521,"REGELS 40712-49775"

49782 DATA 169,55,133,1,96,160,0
49789 DATA 165,252,240,35,177,87,5
49796 DATA 94,145,87,169,132,24,101
49803 DATA 92,133,92,165,253,10,10
49810 DATA 10,10,133,95,177,91,41
49817 DATA 15,24,101,95,145,91,169
49824 DATA 55,133,1,96,165,94,73
49831 DATA 255,133,94,177,87,37,94
49838 DATA 145,87,169,55,133,1,96
49845 DATA 169,4,141,136,2,173,2
49846 DATA 7086,"REGELS 49782-49845"
```

```
49852 DATA 221,41,252,141,2,221,169
49859 DATA 27,141,17,208,169,200,141
49866 DATA 22,208,169,21,141,24,208
49873 DATA 96,32,253,174,32,235,183
49880 DATA 138,141,62,3,169,0,141
49887 DATA 63,3,165,20,141,60,3
49894 DATA 165,21,141,61,3,32,253
49901 DATA 174,32,235,183,138,141,66
49908 DATA 3,169,0,141,67,3,165
49915 DATA 20,141,64,3,165,21,141
49916 DATA 7605,"REGELS 49852-49915"

49922 DATA 65,3,32,253,174,32,235
49929 DATA 183,138,133,252,165,20,133
49936 DATA 254,173,64,3,56,237,60
49943 DATA 3,141,68,3,173,65,3
49950 DATA 237,61,3,141,69,3,173
49957 DATA 66,3,56,237,62,3,141
49964 DATA 70,3,173,63,3,237,67
49971 DATA 3,141,71,3,169,1,141
49978 DATA 94,3,141,96,3,169,0
49985 DATA 141,95,3,141,97,3,173
49986 DATA 6680,"REGELS 49922-49985"

49992 DATA 71,3,41,128,240,8,169
49999 DATA 255,141,94,3,141,95,3
50006 DATA 173,69,3,41,128,240,8
50013 DATA 169,255,141,96,3,141,97
50020 DATA 3,173,69,3,41,128,240
50027 DATA 30,173,69,3,73,255,141
50034 DATA 73,3,24,173,68,3,73
50041 DATA 255,105,1,141,72,3,173
50048 DATA 73,3,105,0,141,73,3
50055 DATA 76,150,195,173,68,3,141
50056 DATA 6728,"REGELS 49992-50055"

50062 DATA 72,3,173,69,3,141,73
50069 DATA 3,173,71,3,41,128,240
50076 DATA 30,173,71,3,73,255,141
50083 DATA 75,3,24,173,70,3,73
50090 DATA 255,105,1,141,74,3,173
50097 DATA 75,3,105,0,141,75,3
50104 DATA 76,199,195,173,70,3,141
50111 DATA 74,3,173,71,3,141,75
50118 DATA 3,173,72,3,56,237,74
50125 DATA 3,141,88,3,173,73,3
50126 DATA 6013,"REGELS 50062-50125"
```

```
50132 DATA 237,75,3,141,89,3,41
50139 DATA 128,240,60,169,255,141,90
50146 DATA 3,141,91,3,169,0,141
50153 DATA 92,3,141,93,3,173,74
50160 DATA 3,141,76,3,173,75,3
50167 DATA 141,77,3,173,72,3,141
50174 DATA 78,3,173,73,3,141,79
50181 DATA 3,173,71,3,41,128,208
50188 DATA 70,169,1,141,90,3,169
50195 DATA 0,141,91,3,76,83,196
50196 DATA 6293,"REGELS 50132-50195"

50202 DATA 169,0,141,90,3,141,91
50209 DATA 3,169,255,141,92,3,141
50216 DATA 93,3,173,72,3,141,76
50223 DATA 3,173,73,3,141,77,3
50230 DATA 173,74,3,141,78,3,173
50237 DATA 75,3,141,79,3,173,69
50244 DATA 3,41,128,208,10,169,1
50251 DATA 141,92,3,169,0,141,93
50258 DATA 3,173,76,3,141,82,3
50265 DATA 173,77,3,141,83,3,173
50266 DATA 5994,"REGELS 50202-50265"

50272 DATA 78,3,141,80,3,173,79
50279 DATA 3,141,81,3,173,76,3
50286 DATA 56,237,78,3,141,84,3
50293 DATA 173,77,3,237,79,3,141
50300 DATA 85,3,78,77,3,110,76
50307 DATA 3,173,78,3,56,237,76
50314 DATA 3,141,86,3,173,79,3
50321 DATA 237,77,3,141,87,3,173
50328 DATA 60,3,133,89,173,61,3
50335 DATA 133,90,173,62,3,133,91
50336 DATA 5826,"REGELS 50272-50335"

50342 DATA 165,254,133,253,32,207,193
50349 DATA 173,87,3,41,128,240,60
50356 DATA 173,86,3,24,109,80,3
50363 DATA 141,86,3,173,87,3,109
50370 DATA 81,3,141,87,3,173,60
50377 DATA 3,24,109,92,3,141,60
50384 DATA 3,173,61,3,109,93,3
50391 DATA 141,61,3,173,62,3,24
50398 DATA 109,90,3,141,62,3,173
50405 DATA 63,3,109,91,3,141,63
50406 DATA 5995,"REGELS 50342-50405"
```

```
50412 DATA 3,76,41,197,173,86,3
50419 DATA 56,237,84,3,141,86,3
50426 DATA 173,87,3,237,85,3,141
50433 DATA 87,3,173,60,3,24,109
50440 DATA 96,3,141,60,3,173,61
50447 DATA 3,109,97,3,141,61,3
50454 DATA 173,62,3,24,109,94,3
50461 DATA 141,62,3,173,63,3,109
50468 DATA 95,3,141,63,3,173,82
50475 DATA 3,56,233,1,141,82,3
50476 DATA 5432,"REGELS 50412-50475"

50482 DATA 173,83,3,233,0,141,83
50489 DATA 3,173,83,3,240,7,201
50496 DATA 255,240,11,76,151,196,173
50503 DATA 82,3,240,3,76,151,196
50510 DATA 96,165,102,208,46,165,101
50517 DATA 208,42,165,100,208,38,165
50524 DATA 99,208,34,165,98,201,128
50531 DATA 240,16,201,192,208,24,165
50538 DATA 97,201,130,208,18,169,3
50545 DATA 133,252,208,30,165,97,240
50546 DATA 9018,"REGELS 50482-50545"

50552 DATA 8,201,129,240,10,201,130
50559 DATA 240,12,32,181,194,76,72
50566 DATA 178,169,1,133,252,208,6
50573 DATA 169,2,133,252,208,0,32
50580 DATA 253,174,32,235,183,134,91
50587 DATA 169,0,133,92,165,20,133
50594 DATA 89,165,21,133,90,32,205
50601 DATA 192,165,251,240,7,165,95
50608 DATA 24,101,94,133,94,169,54
50615 DATA 133,1,160,0,177,87,37
50616 DATA 8397,"REGELS 50552-50615"

50622 DATA 94,240,45,165,251,240,27
50629 DATA 165,94,56,229,95,133,94
50636 DATA 177,87,37,95,240,14,177
50643 DATA 87,37,94,240,16,165,252
50650 DATA 201,3,240,27,208,14,165
50657 DATA 252,201,1,240,19,208,6
50664 DATA 165,252,201,2,240,11,169
50671 DATA 0,141,114,3,133,97,133
50678 DATA 98,240,13,169,1,141,114
50685 DATA 3,169,129,133,97,169,128
50686 DATA 8696,"REGELS 50622-50685"
```

```
50692 DATA 133,98,169,0,170,168,133
50699 DATA 99,133,100,133,101,133,102
50706 DATA 169,55,133,1,169,0,96
50713 DATA 32,253,174,32,235,183,142
50720 DATA 62,3,162,0,32,241,183
50727 DATA 134,254,162,0,32,241,183
50734 DATA 142,115,3,162,0,32,241
50741 DATA 183,141,116,3,169,0,141
50748 DATA 63,3,165,20,141,60,3
50755 DATA 165,21,141,61,3,169,3
50756 DATA 7501,"REGELS 50692-50755"

50762 DATA 205,115,3,48,8,205,116
50769 DATA 3,48,3,76,93,198,32
50776 DATA 181,194,76,72,178,32,70
50783 DATA 199,32,205,192,169,0,133
50790 DATA 2,165,251,208,14,169,1
50797 DATA 141,115,3,141,116,3,141
50804 DATA 106,3,76,126,198,169,2
50811 DATA 141,106,3,169,0,141,118
50818 DATA 3,141,117,3,32,189,199
50825 DATA 32,70,199,173,115,3,133
50826 DATA 7123,"REGELS 50762-50825"

50832 DATA 252,32,167,197,173,114,3
50839 DATA 240,3,76,90,199,32,70
50846 DATA 199,173,116,3,133,252,32
50853 DATA 167,197,173,114,3,240,3
50860 DATA 76,90,199,173,62,3,201
50867 DATA 199,240,42,32,70,199,173
50874 DATA 115,3,133,252,230,91,32
50881 DATA 167,197,173,114,3,208,24
50888 DATA 32,70,199,173,116,3,133
50895 DATA 252,230,91,32,167,197,173
50896 DATA 8822,"REGELS 50832-50895"

50902 DATA 114,3,208,6,32,121,199
50909 DATA 76,229,198,169,0,141,118
50916 DATA 3,169,0,205,62,3,240
50923 DATA 42,32,70,199,173,115,3
50930 DATA 133,252,198,91,32,167,197
50937 DATA 173,114,3,208,24,32,70
50944 DATA 199,173,116,3,133,252,198
50951 DATA 91,32,167,197,173,114,3
50958 DATA 208,6,32,140,199,76,27
50965 DATA 199,169,0,141,117,3,32
50966 DATA 7824,"REGELS 50902-50965"
```

```
50972 DATA 70,199,173,115,3,133,252
50979 DATA 165,254,133,253,32,207,193
50986 DATA 56,173,60,3,237,106,3
50993 DATA 141,60,3,173,61,3,233
51000 DATA 0,141,61,3,201,2,144
51007 DATA 3,76,90,199,76,137,198
51014 DATA 173,60,3,133,89,173,61
51021 DATA 3,133,90,173,62,3,133
51028 DATA 91,169,0,133,92,96,166
51035 DATA 2,240,26,202,189,0,207
51036 DATA 7727,"REGELS 50972-51035"

51042 DATA 141,62,3,202,189,0,207
51049 DATA 141,61,3,202,189,0,207
51056 DATA 141,60,3,134,2,76,126
51063 DATA 198,96,173,118,3,240,1
51070 DATA 96,169,1,141,118,3,32
51077 DATA 70,199,230,91,76,156,199
51084 DATA 173,117,3,240,1,96,169
51091 DATA 1,141,117,3,32,70,199
51098 DATA 198,91,166,2,224,187,144
51105 DATA 6,32,181,194,76,126,185
51106 DATA 7733,"REGELS 51042-51105"

51112 DATA 165,89,157,0,207,232,165
51119 DATA 90,157,0,207,232,165,91
51126 DATA 157,0,207,232,134,2,96
51133 DATA 24,173,60,3,109,106,3
51140 DATA 141,64,3,133,89,173,61
51147 DATA 3,105,0,141,65,3,133
51154 DATA 90,173,62,3,133,91,169
51161 DATA 0,133,92,173,65,3,240
51168 DATA 13,201,1,240,1,96,173
51175 DATA 64,3,201,64,144,1,96
51176 DATA 7102,"REGELS 51112-51175"

51182 DATA 173,116,3,133,252,32,167
51189 DATA 197,173,114,3,240,1,96
51196 DATA 173,64,3,141,60,3,173
51203 DATA 65,3,141,61,3,76,189
51210 DATA 199
51211 DATA 3054,"REGELS 51182-51210",-1
```

Het bovenstaande programma hebben we ingetikt en getest. Alle graphic-routines werkten prima. Het programma is ook direct vanuit het geheugen op de (letterwiel-) printer afgedrukt. Het is dus niet nog eens overgetikt. Alle codes zijn precies zoals ze moeten zijn. Werkt het bij u niet, controleer dan zorgvuldig of alle regels precies zo zijn ingetikt als ze hierboven staan afgedrukt. Controleer zonodig of alle DATA-regels precies zeven codes bevatten; een 0 heeft geen invloed op de som-controle, maar als u de code 0 ergens bent vergeten in te tikken gaat het wel mis!

Wat zijn nu eigenlijk de acht graphic-routines die u zojuist in het geheugen hebt gePOKEd? Welnu, het zijn de volgende:

1. Inschakelen van de Hoge-Resolutiestand met keuze van rand- en afdrukkleur en MODE
2. Van tekstscherm naar grafisch scherm
3. Wissen van het grafische scherm
4. Van grafisch scherm naar tekstscherm
5. Tekenen van een punt met coördinaten x,y in een bepaalde kleur en vorm
6. Tekenen van een lijn in een bepaalde kleur en vorm tussen twee punten x_1,y_1 en x_2,y_2
7. Opvullen (inkleuren) van een gesloten gebied waarin het punt x,y ligt
8. Controle of een punt x,y in een bepaalde vorm getekend is

Voor de programma's in dit boek zijn alleen de routines 1 en 6 van belang.

De routines worden vanuit een BASIC-programma met behulp van de SYS-opdracht aangeroepen. Achter SYS geven we dan het adres van de geheugenplaats waar de eerste machinecode van de desbetreffende machinetaalroutine te vinden is. Daarna kunnen we soms nog één of meer parameters meegeven.

De routines worden als volgt aangeroepen.

```
1.   SYS49152,MODE,KLEUR
           MODE=0: standaard(tweekleuren)instelling
           MODE=1: multi-colourstand
           KLEUR : kleurcode voor de rand en het scherm
                   (rand valt dus weg) (0=zwart; 1=wit)

2.   SYS49182   : tekstscherm naar grafisch scherm
3.   SYS49241   : wissen grafisch scherm
4.   SYS49845   : grafisch scherm naar tekstscherm
```

5. SYS49585,X,Y,KLEUR,VORM
   x : x-coördinaat van het te tekenen punt
   y : y-coördinaat van het te tekenen punt
   KLEUR : afdrukkleur
   VORM : manier waarop punt getekend wordt
   (1 = zichtbaar, 0 = niet zichtbaar, enz.)

6. SYS49874,X1,Y1,X2,Y2,KLEUR,VORM
   X1,Y1 : coördinaten beginpunt
   X2,Y2 : coördinaten eindpunt
   KLEUR : afdrukkleur (0 is bijv. zwart)
   VORM : manier waarop lijn getekend wordt
   (1 = zichtbaar, enz.)

7. SYS50713,X,Y,KLEUR,VORM1,VORM2
   X,Y : coördinaten van een punt in een gesloten gebied
   KLEUR : afdrukkleur
   VORM1,VORM2 : moeten beide 1 zijn

8. USR(VORM),X,Y
   geeft de waarde 1 als het punt x,y in de opgegeven VORM getekend is, anders is de waarde 0.
   USR(1),160,100 geeft 1 als het punt (160,100) zichtbaar is (d.w.z. getekend is in VORM1), anders 0.

U hebt met deze acht graphic-routines veel meer mogelijkheden dan wij in de programma's gebruikt hebben. U kunt, denken wij, vooral met de routine SYS50713,X,Y,1,1,1 zelf naar hartelust vlakken inkleuren. We zullen nu laten zien hoe u de HIRES 0,1 en LINE X1,Y1,X2,Y2,1 opdrachten uit de programma's moet vervangen door SYS-opdrachten.

We nemen aan dat u het in deze appendix gegeven programma hebt ingelezen (of ingetoetst) en gedraaid. De machinetaalroutines bevinden zich nu dus in de geheugenplaatsen 49152 t/m 51210. Nu gaat u het eerste programma uit dit boek intikken. Dit programma herhalen we hier nog even.

```
100 REM PROGRAMMA 1 DIAGONAALWEB
110 PRINT CHR$(147)
120 HIRES 0,1
130 Y1=0 : Y2=200
200 FOR X1=0 TO 320 STEP 40
210 :  FOR X2=0 TO 320 STEP 40
220 :    LINE X1,Y1,X2,Y2,1
230 :  NEXT X2
240 NEXT X1
300 GET A$ : IF A$="" THEN 300
310 END
```

Als u dit programma precies zo intikt als het hier is afgedrukt en als u het draait, krijgt u een foutmelding. Uw Commodore kent de HIRES- en LINE-opdrachten niet. Vervang in bovenstaand programma

```
HIRES 0,1               door   SYS49152,0,1
                         en
LINE X1,Y1,X1,Y2,1      door   SYS49874,X1,Y1,X2,Y2,0,1
```

Het programma ziet er dan zo uit:

```
100 REM PROGRAMMA 1 DIAGONAALWEB
110 PRINT CHR$(147)
120 SYS49152,0,1
130 Y1=0 : Y2=200
200 FOR X1=0 TO 320 STEP 40
210 :  FOR X2=0 TO 320 STEP 40
220 :     SYS49874,X1,Y1,X2,Y2,0,1
230 :  NEXT X2
240 NEXT X1
300 GET A$ : IF A$="" THEN 300
310 END
```

Draai dit programma. U zult zien dat er nog net op het laatst iets misgaat. U krijgt de foutmelding

?ILLEGAL QUANTITY ERROR IN 220

Dit komt omdat de machinetaalroutines er vanuit gaan dat de 320 horizontale beeldschermpunten (de x-coördinaat) een waarde hebben tussen 0 en 319, en de verticale beeldpunten (y-coördinaat) een waarde tussen 0 en 199. Een x van 320 en y van 200 kunnen de routines niet verwerken. Simon's BASIC heeft hier echter geen problemen mee, vandaar de waarde 320 in de FOR-NEXT-opdrachten in de regels 200-240. Veranderen we in deze opdrachten 320 in 319, en 200 in 199, dan zal het programma inderdaad goed draaien!

```
100 REM PROGRAMMA 1 DIAGONAALWEB
110 PRINT CHR$(147)
120 SYS49152,0,1
130 Y1=0 : Y2=199
200 FOR X1=0 TO 319 STEP 40
210 :  FOR X2=0 TO 319 STEP 40
220 :     SYS49874,X1,Y1,X2,Y2,0,1
230 :  NEXT X2
240 NEXT X1
300 GET A$ : IF A$="" THEN 300
305 SYS49845
310 END
```

Tot slot zien we dat we na de GET A$ opdracht in regel 305 hebben opgenomen SYS49845 waardoor na het indrukken van een toets de Commodore overschakelt naar de tekststand (40x25 schermposities). Met RUN/RESTORE krijgt u uw licht-donkerblauwe scherm weer terug.

Verander alle programma's in dit boek op bovenstaande manier en let erop, dat x niet de waarde 320 (of meer) en y niet de waarde 200 (of meer) krijgen. Gebeurt dit wel, pas het programma dan aan, zodat x maximaal 319 kan zijn en y maximaal 199. We geven hieronder nog een voorbeeld van het aangepaste programma 3.

```
100 REM PROGRAMMA 3  DRIEHOEKEN IN PERSPECTIEF
110 PRINT CHR$(147)
120 SYS49152,0,1
130 DIMX(3),Y(3),SX(3),SY(3)
140 FOR J=1 TO 3
150 :  READ SX(J),SY(J)
160 NEXT J
170 X(0)=0 : Y(0)=72
180 FOR K=0 TO 10 STEP 0.5
190 :  FOR J=1 TO 3
200 :     X(J)=X(0)+K*SX(J)
210 :     Y(J)=Y(0)+K*SY(J)
220 :  NEXT J
230 :  SYS49874,X(1),Y(1),X(2),Y(2),0,1
240 :  SYS49874,X(2),Y(2),X(3),Y(3),0,1
250 :  SYS49874,X(3),Y(3),X(1),Y(1),0,1
260 NEXT K
270 GET A$ : IF A$="" THEN 270
275 SYS49845
280 END
290 :    DATA 6,12,20,9,12,-6
```

Bedenk dat, als u uw Commodore uitzet, ook de geheugenplaatsen 49152-51210 gewist worden. Om van de graphic-routines gebruik te kunnen maken zult u het POKE-programma opnieuw moeten inlezen en draaien. Hebt u liever witte lijnen op een zwarte ondergrond, neem dan

SYS49152,0,0 èn
SYS49874,X1,Y1,X2,Y2,1,1

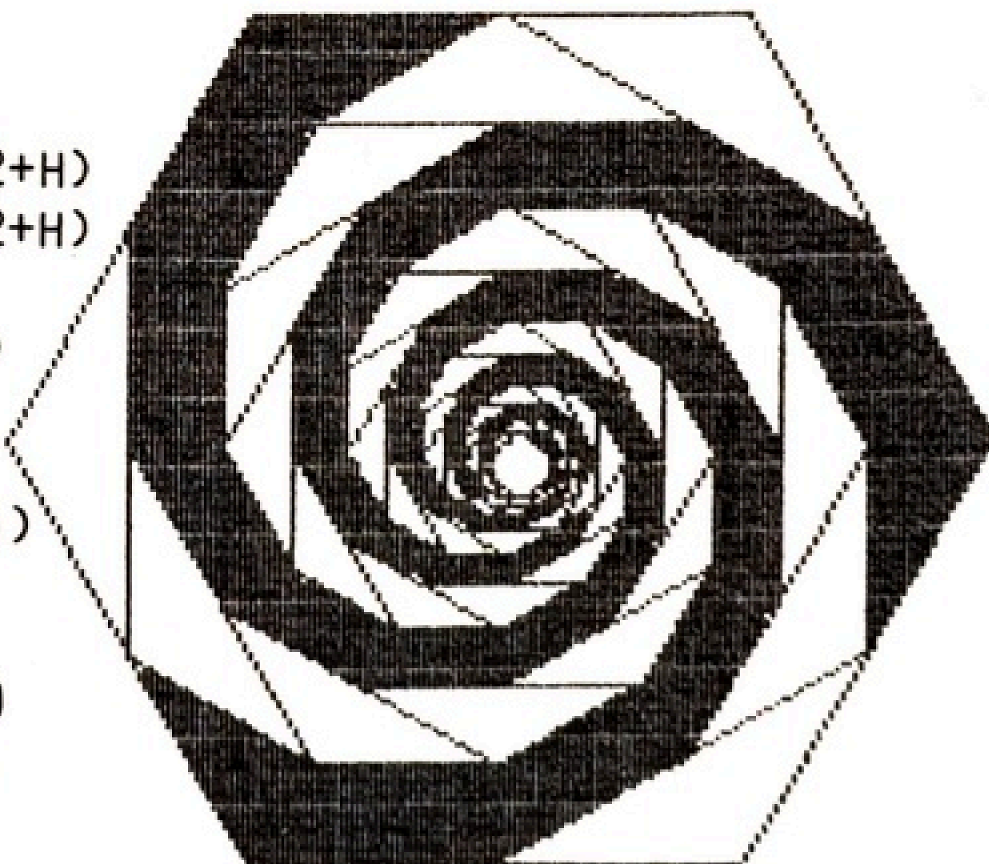
Ook heel mooi is geel op bruin:

SYS49152,0,8 èn
SYS49874,X1,Y1,X2,Y2,7,1

In Appendix 3 geven we een afdruk van de acht machinetaalroutines in Commodore 64 assembleertaal. We hebben deze assemblerlisting gemaakt met een disassembler, die machinecode terugvertaalt in assembleertaalopdrachten.

Hierna volgt nog een voorbeeld van een uitbreiding van programma 4. Hierbij maken we gebruik van het 'inkleuren' van vlakken (regel 270: SYS50713,X,Y,0,1,1). We kiezen steeds drie punten in de drie driehoeken die gevormd worden door een zeshoek en de daarbinnen liggende zeshoek. Het resultaat van het programma zien we in de illustratie.

```
100 REM PROGRAMMA 4 INGESCHREVEN ZESHOEKEN
110 PRINT CHR$(147)
115 REM MACHINETAALROUTINE VOOR HOGE-RESOLUTIESTAND
120 SYS49152,0,1
130 DIM X(6),Y(6),MX(6),MY(6),XO(6),YO(6)
140 U=160:V=100:R=100:H=0.5:W=60*π/180
150 FOR J=0 TO 6
160 :  W1=J*W
170 :  X(J)=INT(U+R*COS(W1)+H)
180 :  Y(J)=INT(V-R*SIN(W1)+H)
190 NEXT J
200 FOR N=1 TO 20
210 :  FOR J=0 TO 5
215 :  REM MACHINETAALROUTINE VOOR LIJN TUSSEN TWEE PUNTEN
220 :    SYS49874,X(J),Y(J),X(J+1),Y(J+1),0,1
230 :  NEXT J
240 IF N=1 THEN 310
245 FOR K = 1 TO 5 STEP 2
250 :  X=INT((X(K)+X(K+1)+XO(K+1))/3)
260 :  Y=INT((Y(K)+Y(K+1)+YO(K+1))/3)
265 :  REM MACHINETAALROUTINE VOOR INKLEUREN
270 :  SYS50713,X,Y,0,1,1
271 NEXT K
310 :  FOR K=0 TO 5
320 :  MX(K)=INT((X(K)+X(K+1))/2+H)
330 :  MY(K)=INT((Y(K)+Y(K+1))/2+H)
340 :  NEXT K
350 :  MX(6)=MX(0) : MY(6)=MY(0)
360 :  FOR J=0 TO 6
370 :     XO(J)=X(J):YO(J)=Y(J)
380 :     X(J)=MX(J) : Y(J)=MY(J)
390 :  NEXT J
400 NEXT N
410 GET A$ : IF A$ = "" THEN 410
414 REM MACHINETAALROUTINE VOOR
    TERUGKEER NAAR TEKSTSCHERM
415 SYS49845
420 END
```



Appendix 3

Acht graphic-routines in Commodore 64 assembleertaal

In de hiernavolgende assembleertaallisting van de in de vorige appendix gegeven machinetaalprogramma's zien we in de eerste kolom de adressen van de geheugenplaatsen, waar de assembleertaalinstructies opgeslagen worden. De eerste assembleertaalinstructie is JSR $AEFD en deze is te vinden vanaf geheugenadres C000. Dit is het hexadecimale equivalent van 49152. De instructie JSR $AEFD wordt in machinetaal gegeven door de hexadecimale codes 20 FD en AE (tweede, derde en vierde kolom in de listing). Deze drie hexadecimale codes 20 FD en AE zijn decimaal 32, 253 en 174. Dit zijn precies de eerste drie machinecodes uit de DATA-regel 49152 uit het programma in Appendix 2. Deze assembleertaallisting laat dus ook de inhoud van de adressen 49152 (C000) t/m 51210 (C808) zien, alleen niet in decimale code maar in hexadecimale code. Assembleertaalprogrammeurs kunnen deze routines wellicht in hun eigen programma's gebruiken of als een voorbeeld van het gebruik van assembleertaalopdrachten.

```
C000  20 FD AE  JSR $AEFD
C003  20 EB B7  JSR $B7EB
C006  8A        TXA
C007  8D 20 D0  STA $D020
C00A  8D 21 D0  STA $D021
C00D  A5 14     LDA $14
C00F  85 FB     STA $FB
C011  F0 02     BEQ $C015
C013  A2 00     LDX #$00
C015  20 59 C0  JSR $C059
C018  20 80 C0  JSR $C080
C01B  20 A6 C0  JSR $C0A6
C01E  A9 3B     LDA #$3B
C020  8D 11 D0  STA $D011
C023  A9 1D     LDA #$1D
C025  8D 18 D0  STA $D018
C028  A5 FB     LDA $FB
C02A  F0 05     BEQ $C031
C02C  A9 D8     LDA #$D8
C02E  8D 16 D0  STA $D016
C031  A9 80     LDA #$80
C033  85 38     STA $38
C035  85 34     STA $34
C037  AD 02 DD  LDA $DD02
C03A  09 03     ORA #$03
C03C  8D 02 DD  STA $DD02
C03F  AD 00 DD  LDA $DD00
C042  29 FC     AND #$FC
C044  09 01     ORA #$01
C046  8D 00 DD  STA $DD00
C049  A9 84     LDA #$84
C04B  8D 88 02  STA $0288
C04E  A9 4F     LDA #$4F
C050  8D 11 03  STA $0311
C053  A9 C5     LDA #$C5
C055  8D 12 03  STA $0312
C058  60        RTS
C059  A0 00     LDY #$00
C05B  A9 40     LDA #$40
C05D  85 57     STA $57
C05F  A9 BF     LDA #$BF
C061  85 58     STA $58
C063  A9 00     LDA #$00
C065  91 57     STA ($57),Y
C067  A5 57     LDA $57
C069  F0 05     BEQ $C070
C06B  C6 57     DEC $57
C06D  4C 63 C0  JMP $C063
C070  C6 58     DEC $58
C072  A5 58     LDA $58
C074  C9 9F     CMP #$9F
C076  F0 07     BEQ $C07F
C078  A9 FF     LDA #$FF
C07A  85 57     STA $57
C07C  4C 63 C0  JMP $C063
C07F  60        RTS
C080  A0 00     LDY #$00
C082  A9 E7     LDA #$E7
C084  85 57     STA $57
C086  A9 87     LDA #$87
C088  85 58     STA $58
C08A  8A        TXA
C08B  91 57     STA ($57),Y
C08D  A5 57     LDA $57
C08F  F0 05     BEQ $C096
C091  C6 57     DEC $57
C093  4C 8A C0  JMP $C08A
C096  C6 58     DEC $58
C098  A5 58     LDA $58
C09A  C9 83     CMP #$83
C09C  F0 07     BEQ $C0A5
C09E  A9 FF     LDA #$FF
C0A0  85 57     STA $57
C0A2  4C 8A C0  JMP $C08A
C0A5  60        RTS
C0A6  A0 00     LDY #$00
C0A8  A9 E7     LDA #$E7
C0AA  85 57     STA $57
C0AC  A9 DB     LDA #$DB
C0AE  85 58     STA $58
C0B0  A9 00     LDA #$00
C0B2  91 57     STA ($57),Y
C0B4  A5 57     LDA $57
C0B6  F0 05     BEQ $C0BD
C0B8  C6 57     DEC $57
C0BA  4C B0 C0  JMP $C0B0
C0BD  C6 58     DEC $58
C0BF  A5 58     LDA $58
C0C1  C9 D7     CMP #$D7
C0C3  F0 07     BEQ $C0CC
```

```
C0C5  A9 FF     LDA #$FF
C0C7  85 57     STA $57
C0C9  4C B0 C0  JMP $C0B0
C0CC  60        RTS
C0CD  A5 5A     LDA $5A
C0CF  C9 00     CMP #$00
C0D1  F0 10     BEQ $C0E3
C0D3  C9 01     CMP #$01
C0D5  D0 06     BNE $C0DD
C0D7  A5 59     LDA $59
C0D9  C9 40     CMP #$40
C0DB  90 06     BCC $C0E3
C0DD  20 B5 C2  JSR $C2B5
C0E0  4C 48 B2  JMP $B248
C0E3  A5 5B     LDA $5B
C0E5  C9 C8     CMP #$C8
C0E7  90 06     BCC $C0EF
C0E9  20 B5 C2  JSR $C2B5
C0EC  4C 48 B2  JMP $B248
C0EF  A5 59     LDA $59
C0F1  29 07     AND #$07
C0F3  85 5E     STA $5E
C0F5  A9 07     LDA #$07
C0F7  38        SEC
C0F8  E5 5E     SBC $5E
C0FA  85 5E     STA $5E
C0FC  A5 FB     LDA $FB
C0FE  F0 0B     BEQ $C10B
C100  46 5E     LSR $5E
C102  06 5E     ASL $5E
C104  A5 5E     LDA $5E
C106  18        CLC
C107  69 01     ADC #$01
C109  85 5F     STA $5F
C10B  A5 5E     LDA $5E
C10D  F0 09     BEQ $C118
C10F  A8        TAY
C110  A9 01     LDA #$01
C112  0A        ASL
C113  88        DEY
C114  D0 FC     BNE $C112
C116  F0 02     BEQ $C11A
C118  A9 01     LDA #$01
C11A  85 5E     STA $5E
C11C  A5 5F     LDA $5F
C11E  F0 09     BEQ $C129
C120  A8        TAY
C121  A9 01     LDA #$01
C123  0A        ASL
C124  88        DEY
C125  D0 FC     BNE $C123
C127  F0 02     BEQ $C12B
C129  A9 01     LDA #$01
C12B  85 5F     STA $5F
C12D  A9 00     LDA #$00
C12F  85 5C     STA $5C
C131  85 58     STA $58
C133  85 57     STA $57
C135  A5 5B     LDA $5B
C137  29 07     AND #$07
C139  85 5D     STA $5D
C13B  A5 5B     LDA $5B
C13D  4A        LSR
C13E  4A        LSR
C13F  4A        LSR
C140  85 5B     STA $5B
C142  A0 05     LDY #$05
C144  18        CLC
C145  0A        ASL
C146  26 58     ROL $58
C148  88        DEY
C149  D0 F9     BNE $C144
C14B  85 57     STA $57
C14D  A5 5B     LDA $5B
C14F  A0 03     LDY #$03
C151  18        CLC
C152  0A        ASL
C153  88        DEY
C154  D0 FB     BNE $C151
C156  85 5B     STA $5B
C158  18        CLC
C159  65 57     ADC $57
C15B  85 5B     STA $5B
C15D  A5 58     LDA $58
C15F  69 00     ADC #$00
C161  85 5C     STA $5C
C163  A0 03     LDY #$03
C165  18        CLC
C166  46 5A     LSR $5A
C168  66 59     ROR $59
```

```
C16A  88         DEY
C16B  D0 F8      BNE $C165
C16D  A5 59      LDA $59
C16F  85 57      STA $57
C171  A5 5A      LDA $5A
C173  85 58      STA $58
C175  A0 03      LDY #$03
C177  18         CLC
C178  06 57      ASL $57
C17A  26 58      ROL $58
C17C  88         DEY
C17D  D0 F8      BNE $C177
C17F  A0 08      LDY #$08
C181  18         CLC
C182  A5 5B      LDA $5B
C184  65 57      ADC $57
C186  85 57      STA $57
C188  A5 5C      LDA $5C
C18A  65 58      ADC $58
C18C  85 58      STA $58
C18E  88         DEY
C18F  D0 F0      BNE $C181
C191  18         CLC
C192  A5 5D      LDA $5D
C194  65 57      ADC $57
C196  85 57      STA $57
C198  A9 00      LDA #$00
C19A  65 58      ADC $58
C19C  18         CLC
C19D  A9 A0      LDA #$A0
C19F  65 58      ADC $58
C1A1  85 58      STA $58
C1A3  18         CLC
C1A4  A5 59      LDA $59
C1A6  65 5B      ADC $5B
C1A8  85 5B      STA $5B
C1AA  A9 00      LDA #$00
C1AC  65 5C      ADC $5C
C1AE  85 5C      STA $5C
C1B0  60         RTS
C1B1  20 FD AE   JSR $AEFD
C1B4  20 EB B7   JSR $B7EB
C1B7  A5 14      LDA $14
C1B9  85 59      STA $59
C1BB  A5 15      LDA $15
C1BD  85 5A      STA $5A
C1BF  8A         TXA
C1C0  85 5B      STA $5B
C1C2  20 FD AE   JSR $AEFD
C1C5  20 EB B7   JSR $B7EB
C1C8  8A         TXA
C1C9  85 FC      STA $FC
C1CB  A5 14      LDA $14
C1CD  85 FD      STA $FD
C1CF  20 CD C0   JSR $C0CD
C1D2  A9 36      LDA #$36
C1D4  85 01      STA $01
C1D6  A5 FB      LDA $FB
C1D8  D0 03      BNE $C1DD
C1DA  4C 7B C2   JMP $C27B
C1DD  A5 FC      LDA $FC
C1DF  C9 00      CMP #$00
C1E1  F0 11      BEQ $C1F4
C1E3  C9 01      CMP #$01
C1E5  F0 28      BEQ $C20F
C1E7  C9 02      CMP #$02
C1E9  F0 51      BEQ $C23C
C1EB  C9 03      CMP #$03
C1ED  F0 72      BEQ $C261
C1EF  A9 37      LDA #$37
C1F1  85 01      STA $01
C1F3  60         RTS
C1F4  A0 00      LDY #$00
C1F6  A5 5E      LDA $5E
C1F8  49 FF      EOR #$FF
C1FA  85 5E      STA $5E
C1FC  A5 5F      LDA $5F
C1FE  49 FF      EOR #$FF
C200  85 5F      STA $5F
C202  B1 57      LDA ($57),Y
C204  25 5E      AND $5E
C206  25 5F      AND $5F
C208  91 57      STA ($57),Y
C20A  A9 37      LDA #$37
C20C  85 01      STA $01
C20E  60         RTS
C20F  A0 00      LDY #$00
C211  A5 5F      LDA $5F
C213  49 FF      EOR #$FF
C215  85 5F      STA $5F
```

```
C217  B1 57     LDA ($57),Y
C219  05 5E     ORA $5E
C21B  25 5F     AND $5F
C21D  91 57     STA ($57),Y
C21F  A9 84     LDA #$84
C221  18        CLC
C222  65 5C     ADC $5C
C224  85 5C     STA $5C
C226  06 FD     ASL $FD
C228  06 FD     ASL $FD
C22A  06 FD     ASL $FD
C22C  06 FD     ASL $FD
C22E  B1 5B     LDA ($5B),Y
C230  29 0F     AND #$0F
C232  18        CLC
C233  65 FD     ADC $FD
C235  91 5B     STA ($5B),Y
C237  A9 37     LDA #$37
C239  85 01     STA $01
C23B  60        RTS
C23C  A0 00     LDY #$00
C23E  A5 5E     LDA $5E
C240  49 FF     EOR #$FF
C242  85 5E     STA $5E
C244  B1 57     LDA ($57),Y
C246  25 5E     AND $5E
C248  05 5F     ORA $5F
C24A  91 57     STA ($57),Y
C24C  18        CLC
C24D  A9 84     LDA #$84
C24F  65 5C     ADC $5C
C251  85 5C     STA $5C
C253  B1 5B     LDA ($5B),Y
C255  29 F0     AND #$F0
C257  18        CLC
C258  65 FD     ADC $FD
C25A  91 5B     STA ($5B),Y
C25C  A9 37     LDA #$37
C25E  85 01     STA $01
C260  60        RTS
C261  A0 00     LDY #$00
C263  B1 57     LDA ($57),Y
C265  05 5E     ORA $5E
C267  05 5F     ORA $5F
C269  91 57     STA ($57),Y
C26B  A9 D8     LDA #$D8
C26D  18        CLC
C26E  65 5C     ADC $5C
C270  85 5C     STA $5C
C272  A5 FD     LDA $FD
C274  91 5B     STA ($5B),Y
C276  A9 37     LDA #$37
C278  85 01     STA $01
C27A  60        RTS
C27B  A0 00     LDY #$00
C27D  A5 FC     LDA $FC
C27F  F0 23     BEQ $C2A4
C281  B1 57     LDA ($57),Y
C283  05 5E     ORA $5E
C285  91 57     STA ($57),Y
C287  A9 84     LDA #$84
C289  18        CLC
C28A  65 5C     ADC $5C
C28C  85 5C     STA $5C
C28E  A5 FD     LDA $FD
C290  0A        ASL
C291  0A        ASL
C292  0A        ASL
C293  0A        ASL
C294  85 5F     STA $5F
C296  B1 5B     LDA ($5B),Y
C298  29 0F     AND #$0F
C29A  18        CLC
C29B  65 5F     ADC $5F
C29D  91 5B     STA ($5B),Y
C29F  A9 37     LDA #$37
C2A1  85 01     STA $01
C2A3  60        RTS
C2A4  A5 5E     LDA $5E
C2A6  49 FF     EOR #$FF
C2A8  85 5E     STA $5E
C2AA  B1 57     LDA ($57),Y
C2AC  25 5E     AND $5E
C2AE  91 57     STA ($57),Y
C2B0  A9 37     LDA #$37
C2B2  85 01     STA $01
C2B4  60        RTS
C2B5  A9 04     LDA #$04
C2B7  8D 88 02  STA $0288
C2BA  AD 02 DD  LDA $DD02
```

```
C2BD   29 FC      AND #$FC
C2BF   8D 02 DD   STA $DD02
C2C2   A9 1B      LDA #$1B
C2C4   8D 11 D0   STA $D011
C2C7   A9 C8      LDA #$C8
C2C9   8D 16 D0   STA $D016
C2CC   A9 15      LDA #$15
C2CE   8D 18 D0   STA $D018
C2D1   60         RTS
C2D2   20 FD AE   JSR $AEFD
C2D5   20 EB B7   JSR $B7EB
C2D8   8A         TXA
C2D9   8D 3E 03   STA $033E
C2DC   A9 00      LDA #$00
C2DE   8D 3F 03   STA $033F
C2E1   A5 14      LDA $14
C2E3   8D 3C 03   STA $033C
C2E6   A5 15      LDA $15
C2E8   8D 3D 03   STA $033D
C2EB   20 FD AE   JSR $AEFD
C2EE   20 EB B7   JSR $B7EB
C2F1   8A         TXA
C2F2   8D 42 03   STA $0342
C2F5   A9 00      LDA #$00
C2F7   8D 43 03   STA $0343
C2FA   A5 14      LDA $14
C2FC   8D 40 03   STA $0340
C2FF   A5 15      LDA $15
C301   8D 41 03   STA $0341
C304   20 FD AE   JSR $AEFD
C307   20 EB B7   JSR $B7EB
C30A   8A         TXA
C30B   85 FC      STA $FC
C30D   A5 14      LDA $14
C30F   85 FE      STA $FE
C311   AD 40 03   LDA $0340
C314   38         SEC
C315   ED 3C 03   SBC $033C
C318   8D 44 03   STA $0344
C31B   AD 41 03   LDA $0341
C31E   ED 3D 03   SBC $033D
C321   8D 45 03   STA $0345
C324   AD 42 03   LDA $0342
C327   38         SEC
C328   ED 3E 03   SBC $033E
C32B   8D 46 03   STA $0346
C32E   AD 3F 03   LDA $033F
C331   ED 43 03   SBC $0343
C334   8D 47 03   STA $0347
C337   A9 01      LDA #$01
C339   8D 5E 03   STA $035E
C33C   8D 60 03   STA $0360
C33F   A9 00      LDA #$00
C341   8D 5F 03   STA $035F
C344   8D 61 03   STA $0361
C347   AD 47 03   LDA $0347
C34A   29 80      AND #$80
C34C   F0 08      BEQ $C356
C34E   A9 FF      LDA #$FF
C350   8D 5E 03   STA $035E
C353   8D 5F 03   STA $035F
C356   AD 45 03   LDA $0345
C359   29 80      AND #$80
C35B   F0 08      BEQ $C365
C35D   A9 FF      LDA #$FF
C35F   8D 60 03   STA $0360
C362   8D 61 03   STA $0361
C365   AD 45 03   LDA $0345
C368   29 80      AND #$80
C36A   F0 1E      BEQ $C38A
C36C   AD 45 03   LDA $0345
C36F   49 FF      EOR #$FF
C371   8D 49 03   STA $0349
C374   18         CLC
C375   AD 44 03   LDA $0344
C378   49 FF      EOR #$FF
C37A   69 01      ADC #$01
C37C   8D 48 03   STA $0348
C37F   AD 49 03   LDA $0349
C382   69 00      ADC #$00
C384   8D 49 03   STA $0349
C387   4C 96 C3   JMP $C396
C38A   AD 44 03   LDA $0344
C38D   8D 48 03   STA $0348
C390   AD 45 03   LDA $0345
C393   8D 49 03   STA $0349
C396   AD 47 03   LDA $0347
C399   29 80      AND #$80
C39B   F0 1E      BEQ $C3BB
C39D   AD 47 03   LDA $0347
```

```
C3A0  49 FF     EOR #$FF
C3A2  8D 4B 03  STA $034B
C3A5  18        CLC
C3A6  AD 46 03  LDA $0346
C3A9  49 FF     EOR #$FF
C3AB  69 01     ADC #$01
C3AD  8D 4A 03  STA $034A
C3B0  AD 4B 03  LDA $034B
C3B3  69 00     ADC #$00
C3B5  8D 4B 03  STA $034B
C3B8  4C C7 C3  JMP $C3C7
C3BB  AD 46 03  LDA $0346
C3BE  8D 4A 03  STA $034A
C3C1  AD 47 03  LDA $0347
C3C4  8D 4B 03  STA $034B
C3C7  AD 48 03  LDA $0348
C3CA  38        SEC
C3CB  ED 4A 03  SBC $034A
C3CE  8D 58 03  STA $0358
C3D1  AD 49 03  LDA $0349
C3D4  ED 4B 03  SBC $034B
C3D7  8D 59 03  STA $0359
C3DA  29 80     AND #$80
C3DC  F0 3C     BEQ $C41A
C3DE  A9 FF     LDA #$FF
C3E0  8D 5A 03  STA $035A
C3E3  8D 5B 03  STA $035B
C3E6  A9 00     LDA #$00
C3E8  8D 5C 03  STA $035C
C3EB  8D 5D 03  STA $035D
C3EE  AD 4A 03  LDA $034A
C3F1  8D 4C 03  STA $034C
C3F4  AD 4B 03  LDA $034B
C3F7  8D 4D 03  STA $034D
C3FA  AD 48 03  LDA $0348
C3FD  8D 4E 03  STA $034E
C400  AD 49 03  LDA $0349
C403  8D 4F 03  STA $034F
C406  AD 47 03  LDA $0347
C409  29 80     AND #$80
C40B  D0 46     BNE $C453
C40D  A9 01     LDA #$01
C40F  8D 5A 03  STA $035A
C412  A9 00     LDA #$00
C414  8D 5B 03  STA $035B
C417  4C 53 C4  JMP $C453
C41A  A9 00     LDA #$00
C41C  8D 5A 03  STA $035A
C41F  8D 5B 03  STA $035B
C422  A9 FF     LDA #$FF
C424  8D 5C 03  STA $035C
C427  8D 5D 03  STA $035D
C42A  AD 48 03  LDA $0348
C42D  8D 4C 03  STA $034C
C430  AD 49 03  LDA $0349
C433  8D 4D 03  STA $034D
C436  AD 4A 03  LDA $034A
C439  8D 4E 03  STA $034E
C43C  AD 4B 03  LDA $034B
C43F  8D 4F 03  STA $034F
C442  AD 45 03  LDA $0345
C445  29 80     AND #$80
C447  D0 0A     BNE $C453
C449  A9 01     LDA #$01
C44B  8D 5C 03  STA $035C
C44E  A9 00     LDA #$00
C450  8D 5D 03  STA $035D
C453  AD 4C 03  LDA $034C
C456  8D 52 03  STA $0352
C459  AD 4D 03  LDA $034D
C45C  8D 53 03  STA $0353
C45F  AD 4E 03  LDA $034E
C462  8D 50 03  STA $0350
C465  AD 4F 03  LDA $034F
C468  8D 51 03  STA $0351
C46B  AD 4C 03  LDA $034C
C46E  38        SEC
C46F  ED 4E 03  SBC $034E
C472  8D 54 03  STA $0354
C475  AD 4D 03  LDA $034D
C478  ED 4F 03  SBC $034F
C47B  8D 55 03  STA $0355
C47E  4E 4D 03  LSR $034D
C481  6E 4C 03  ROR $034C
C484  AD 4E 03  LDA $034E
C487  38        SEC
C488  ED 4C 03  SBC $034C
C48B  8D 56 03  STA $0356
C48E  AD 4F 03  LDA $034F
C491  ED 4D 03  SBC $034D
```

```
C494   8D 57 03 STA $0357
C497   AD 3C 03 LDA $033C
C49A   85 59    STA $59
C49C   AD 3D 03 LDA $033D
C49F   85 5A    STA $5A
C4A1   AD 3E 03 LDA $033E
C4A4   85 5B    STA $5B
C4A6   A5 FE    LDA $FE
C4A8   85 FD    STA $FD
C4AA   20 CF C1 JSR $C1CF
C4AD   AD 57 03 LDA $0357
C4B0   29 80    AND #$80
C4B2   F0 3C    BEQ $C4F0
C4B4   AD 56 03 LDA $0356
C4B7   18       CLC
C4B8   6D 50 03 ADC $0350
C4BB   8D 56 03 STA $0356
C4BE   AD 57 03 LDA $0357
C4C1   6D 51 03 ADC $0351
C4C4   8D 57 03 STA $0357
C4C7   AD 3C 03 LDA $033C
C4CA   18       CLC
C4CB   6D 5C 03 ADC $035C
C4CE   8D 3C 03 STA $033C
C4D1   AD 3D 03 LDA $033D
C4D4   6D 5D 03 ADC $035D
C4D7   8D 3D 03 STA $033D
C4DA   AD 3E 03 LDA $033E
C4DD   18       CLC
C4DE   6D 5A 03 ADC $035A
C4E1   8D 3E 03 STA $033E
C4E4   AD 3F 03 LDA $033F
C4E7   6D 5B 03 ADC $035B
C4EA   8D 3F 03 STA $033F
C4ED   4C 29 C5 JMP $C529
C4F0   AD 56 03 LDA $0356
C4F3   38       SEC
C4F4   ED 54 03 SBC $0354
C4F7   8D 56 03 STA $0356
C4FA   AD 57 03 LDA $0357
C4FD   ED 55 03 SBC $0355
C500   8D 57 03 STA $0357
C503   AD 3C 03 LDA $033C
C506   18       CLC
C507   6D 60 03 ADC $0360
C50A   8D 3C 03 STA $033C
C50D   AD 3D 03 LDA $033D
C510   6D 61 03 ADC $0361
C513   8D 3D 03 STA $033D
C516   AD 3E 03 LDA $033E
C519   18       CLC
C51A   6D 5E 03 ADC $035E
C51D   8D 3E 03 STA $033E
C520   AD 3F 03 LDA $033F
C523   6D 5F 03 ADC $035F
C526   8D 3F 03 STA $033F
C529   AD 52 03 LDA $0352
C52C   38       SEC
C52D   E9 01    SBC #$01
C52F   8D 52 03 STA $0352
C532   AD 53 03 LDA $0353
C535   E9 00    SBC #$00
C537   8D 53 03 STA $0353
C53A   AD 53 03 LDA $0353
C53D   F0 07    BEQ $C546
C53F   C9 FF    CMP #$FF
C541   F0 0B    BEQ $C54E
C543   4C 97 C4 JMP $C497
C546   AD 52 03 LDA $0352
C549   F0 03    BEQ $C54E
C54B   4C 97 C4 JMP $C497
C54E   60       RTS
C54F   A5 66    LDA $66
C551   D0 2E    BNE $C581
C553   A5 65    LDA $65
C555   D0 2A    BNE $C581
C557   A5 64    LDA $64
C559   D0 26    BNE $C581
C55B   A5 63    LDA $63
C55D   D0 22    BNE $C581
C55F   A5 62    LDA $62
C561   C9 80    CMP #$80
C563   F0 10    BEQ $C575
C565   C9 C0    CMP #$C0
C567   D0 18    BNE $C581
C569   A5 61    LDA $61
C56B   C9 82    CMP #$82
C56D   D0 12    BNE $C581
C56F   A9 03    LDA #$03
C571   85 FC    STA $FC
C573   D0 1E    BNE $C593
C575   A5 61    LDA $61
```

```
C577  F0 08     BEQ $C581
C579  C9 81     CMP #$81
C57B  F0 0A     BEQ $C587
C57D  C9 82     CMP #$82
C57F  F0 0C     BEQ $C58D
C581  20 B5 C2  JSR $C2B5
C584  4C 48 B2  JMP $B248
C587  A9 01     LDA #$01
C589  85 FC     STA $FC
C58B  D0 06     BNE $C593
C58D  A9 02     LDA #$02
C58F  85 FC     STA $FC
C591  D0 00     BNE $C593
C593  20 FD AE  JSR $AEFD
C596  20 EB B7  JSR $B7EB
C599  86 5B     STX $5B
C59B  A9 00     LDA #$00
C59D  85 5C     STA $5C
C59F  A5 14     LDA $14
C5A1  85 59     STA $59
C5A3  A5 15     LDA $15
C5A5  85 5A     STA $5A
C5A7  20 CD C0  JSR $C0CD
C5AA  A5 FB     LDA $FB
C5AC  F0 07     BEQ $C5B5
C5AE  A5 5F     LDA $5F
C5B0  18        CLC
C5B1  65 5E     ADC $5E
C5B3  85 5E     STA $5E
C5B5  A9 36     LDA #$36
C5B7  85 01     STA $01
C5B9  A0 00     LDY #$00
C5BB  B1 57     LDA ($57),Y
C5BD  25 5E     AND $5E
C5BF  F0 2D     BEQ $C5EE
C5C1  A5 FB     LDA $FB
C5C3  F0 1B     BEQ $C5E0
C5C5  A5 5E     LDA $5E
C5C7  38        SEC
C5C8  E5 5F     SBC $5F
C5CA  85 5E     STA $5E
C5CC  B1 57     LDA ($57),Y
C5CE  25 5F     AND $5F
C5D0  F0 0E     BEQ $C5E0
C5D2  B1 57     LDA ($57),Y
C5D4  25 5E     AND $5E
C5D6  F0 10     BEQ $C5E8
C5D8  A5 FC     LDA $FC
C5DA  C9 03     CMP #$03
C5DC  F0 1B     BEQ $C5F9
C5DE  D0 0E     BNE $C5EE
C5E0  A5 FC     LDA $FC
C5E2  C9 01     CMP #$01
C5E4  F0 13     BEQ $C5F9
C5E6  D0 06     BNE $C5EE
C5E8  A5 FC     LDA $FC
C5EA  C9 02     CMP #$02
C5EC  F0 0B     BEQ $C5F9
C5EE  A9 00     LDA #$00
C5F0  8D 72 03  STA $0372
C5F3  85 61     STA $61
C5F5  85 62     STA $62
C5F7  F0 0D     BEQ $C606
C5F9  A9 01     LDA #$01
C5FB  8D 72 03  STA $0372
C5FE  A9 81     LDA #$81
C600  85 61     STA $61
C602  A9 80     LDA #$80
C604  85 62     STA $62
C606  A9 00     LDA #$00
C608  AA        TAX
C609  A8        TAY
C60A  85 63     STA $63
C60C  85 64     STA $64
C60E  85 65     STA $65
C610  85 66     STA $66
C612  A9 37     LDA #$37
C614  85 01     STA $01
C616  A9 00     LDA #$00
C618  60        RTS
C619  20 FD AE  JSR $AEFD
C61C  20 EB B7  JSR $B7EB
C61F  8E 3E 03  STX $033E
C622  A2 00     LDX #$00
C624  20 F1 B7  JSR $B7F1
C627  86 FE     STX $FE
C629  A2 00     LDX #$00
C62B  20 F1 B7  JSR $B7F1
C62E  8E 73 03  STX $0373
C631  A2 00     LDX #$00
```

```
C633  20 F1 B7 JSR $B7F1
C636  8D 74 03 STA $0374
C639  A9 00    LDA #$00
C63B  8D 3F 03 STA $033F
C63E  A5 14    LDA $14
C640  8D 3C 03 STA $033C
C643  A5 15    LDA $15
C645  8D 3D 03 STA $033D
C648  A9 03    LDA #$03
C64A  CD 73 03 CMP $0373
C64D  30 08    BMI $C657
C64F  CD 74 03 CMP $0374
C652  30 03    BMI $C657
C654  4C 5D C6 JMP $C65D
C657  20 B5 C2 JSR $C2B5
C65A  4C 48 B2 JMP $B248
C65D  20 46 C7 JSR $C746
C660  20 CD C0 JSR $C0CD
C663  A9 00    LDA #$00
C665  85 02    STA $02
C667  A5 FB    LDA $FB
C669  D0 0E    BNE $C679
C66B  A9 01    LDA #$01
C66D  8D 73 03 STA $0373
C670  8D 74 03 STA $0374
C673  8D 6A 03 STA $036A
C676  4C 7E C6 JMP $C67E
C679  A9 02    LDA #$02
C67B  8D 6A 03 STA $036A
C67E  A9 00    LDA #$00
C680  8D 76 03 STA $0376
C683  8D 75 03 STA $0375
C686  20 BD C7 JSR $C7BD
C689  20 46 C7 JSR $C746
C68C  AD 73 03 LDA $0373
C68F  85 FC    STA $FC
C691  20 A7 C5 JSR $C5A7
C694  AD 72 03 LDA $0372
C697  F0 03    BEQ $C69C
C699  4C 5A C7 JMP $C75A
C69C  20 46 C7 JSR $C746
C69F  AD 74 03 LDA $0374
C6A2  85 FC    STA $FC
C6A4  20 A7 C5 JSR $C5A7
C6A7  AD 72 03 LDA $0372
C6AA  F0 03    BEQ $C6AF
C6AC  4C 5A C7 JMP $C75A
C6AF  AD 3E 03 LDA $033E
C6B2  C9 C7    CMP #$C7
C6B4  F0 2A    BEQ $C6E0
C6B6  20 46 C7 JSR $C746
C6B9  AD 73 03 LDA $0373
C6BC  85 FC    STA $FC
C6BE  E6 5B    INC $5B
C6C0  20 A7 C5 JSR $C5A7
C6C3  AD 72 03 LDA $0372
C6C6  D0 18    BNE $C6E0
C6C8  20 46 C7 JSR $C746
C6CB  AD 74 03 LDA $0374
C6CE  85 FC    STA $FC
C6D0  E6 5B    INC $5B
C6D2  20 A7 C5 JSR $C5A7
C6D5  AD 72 03 LDA $0372
C6D8  D0 06    BNE $C6E0
C6DA  20 79 C7 JSR $C779
C6DD  4C E5 C6 JMP $C6E5
C6E0  A9 00    LDA #$00
C6E2  8D 76 03 STA $0376
C6E5  A9 00    LDA #$00
C6E7  CD 3E 03 CMP $033E
C6EA  F0 2A    BEQ $C716
C6EC  20 46 C7 JSR $C746
C6EF  AD 73 03 LDA $0373
C6F2  85 FC    STA $FC
C6F4  C6 5B    DEC $5B
C6F6  20 A7 C5 JSR $C5A7
C6F9  AD 72 03 LDA $0372
C6FC  D0 18    BNE $C716
C6FE  20 46 C7 JSR $C746
C701  AD 74 03 LDA $0374
C704  85 FC    STA $FC
C706  C6 5B    DEC $5B
C708  20 A7 C5 JSR $C5A7
C70B  AD 72 03 LDA $0372
C70E  D0 06    BNE $C716
C710  20 8C C7 JSR $C78C
C713  4C 1B C7 JMP $C71B
C716  A9 00    LDA #$00
C718  8D 75 03 STA $0375
C71B  20 46 C7 JSR $C746
```

```
C71E   AD 73 03 LDA $0373
C721   85 FC    STA $FC
C723   A5 FE    LDA $FE
C725   85 FD    STA $FD
C727   20 CF C1 JSR $C1CF
C72A   38       SEC
C72B   AD 3C 03 LDA $033C
C72E   ED 6A 03 SBC $036A
C731   8D 3C 03 STA $033C
C734   AD 3D 03 LDA $033D
C737   E9 00    SBC #$00
C739   8D 3D 03 STA $033D
C73C   C9 02    CMP #$02
C73E   90 03    BCC $C743
C740   4C 5A C7 JMP $C75A
C743   4C 89 C6 JMP $C689
C746   AD 3C 03 LDA $033C
C749   85 59    STA $59
C74B   AD 3D 03 LDA $033D
C74E   85 5A    STA $5A
C750   AD 3E 03 LDA $033E
C753   85 5B    STA $5B
C755   A9 00    LDA #$00
C757   85 5C    STA $5C
C759   60       RTS
C75A   A6 02    LDX $02
C75C   F0 1A    BEQ $C778
C75E   CA       DEX
C75F   BD 00 CF LDA $CF00,X
C762   8D 3E 03 STA $033E
C765   CA       DEX
C766   BD 00 CF LDA $CF00,X
C769   8D 3D 03 STA $033D
C76C   CA       DEX
C76D   BD 00 CF LDA $CF00,X
C770   8D 3C 03 STA $033C
C773   86 02    STX $02
C775   4C 7E C6 JMP $C67E
C778   60       RTS
C779   AD 76 03 LDA $0376
C77C   F0 01    BEQ $C77F
C77E   60       RTS
C77F   A9 01    LDA #$01
C781   8D 76 03 STA $0376
C784   20 46 C7 JSR $C746
C787   E6 5B    INC $5B
C789   4C 9C C7 JMP $C79C
C78C   AD 75 03 LDA $0375
C78F   F0 01    BEQ $C792
C791   60       RTS
C792   A9 01    LDA #$01
```

```
C794   8D 75 03 STA $0375
C797   20 46 C7 JSR $C746
C79A   C6 5B    DEC $5B
C79C   A6 02    LDX $02
C79E   E0 BB    CPX #$BB
C7A0   90 06    BCC $C7A8
C7A2   20 B5 C2 JSR $C2B5
C7A5   4C 7E B9 JMP $B97E
C7A8   A5 59    LDA $59
C7AA   9D 00 CF STA $CF00,X
C7AD   E8       INX
C7AE   A5 5A    LDA $5A
C7B0   9D 00 CF STA $CF00,X
C7B3   E8       INX
C7B4   A5 5B    LDA $5B
C7B6   9D 00 CF STA $CF00,X
C7B9   E8       INX
C7BA   86 02    STX $02
C7BC   60       RTS
C7BD   18       CLC
C7BE   AD 3C 03 LDA $033C
C7C1   6D 6A 03 ADC $036A
C7C4   8D 40 03 STA $0340
C7C7   85 59    STA $59
C7C9   AD 3D 03 LDA $033D
C7CC   69 00    ADC #$00
C7CE   8D 41 03 STA $0341
C7D1   85 5A    STA $5A
C7D3   AD 3E 03 LDA $033E
C7D6   85 5B    STA $5B
C7D8   A9 00    LDA #$00
C7DA   85 5C    STA $5C
C7DC   AD 41 03 LDA $0341
C7DF   F0 0D    BEQ $C7EE
C7E1   C9 01    CMP #$01
C7E3   F0 01    BEQ $C7E6
C7E5   60       RTS
C7E6   AD 40 03 LDA $0340
C7E9   C9 40    CMP #$40
C7EB   90 01    BCC $C7EE
C7ED   60       RTS
C7EE   AD 74 03 LDA $0374
C7F1   85 FC    STA $FC
C7F3   20 A7 C5 JSR $C5A7
C7F6   AD 72 03 LDA $0372
C7F9   F0 01    BEQ $C7FC
C7FB   60       RTS
C7FC   AD 40 03 LDA $0340
C7FF   8D 3C 03 STA $033C
C802   AD 41 03 LDA $0341
C805   8D 3D 03 STA $033D
C808   4C BD C7 JMP $C7BD
```

ACADEMIC SERVICE INFORMATICA UITGAVEN

INLEIDINGEN

Computers en onze samenleving van M.A. Arbib
Basiskennis informatieverwerking van Jan Everink
De viewdata revolutie van S. Fedida en R. Malik
De informatiemaatschappij van Jan Everink
Informatica, een theoretische inleiding van dr. L.P.J. Groenewegen en prof.dr. A. Ollongren
AIV, Automatisering van de informatieverzorging van ir. Th.J.G. Derksen en drs. H.W. Crins
Organisatie, informatie en computers van David M. Kroenke

MICROCOMPUTERS

Programmeercursus Microsoft BASIC van Nok van Veen
Werken met bestanden in BASIC van L. Finkel en J.R. Brown
Werken met bestanden op de Apple van L. Finkel en J.R. Brown
Werken met Visicalc van C. Klitzner en M.J. Plociak, Jr.
CP/M: een gids voor zelfstudie van J.N. Fernandez en R. Ashley
Cursus Z-80 assembleertaal van Roger Hutty
Exidy sorcerer en BASIC van Nok van Veen e.a.
TRS-80 BASIC: een gids voor zelfstudie van Bob Albrecht e.a.
TRS-80 BASIC voor gevorderden van Don Inman e.a.
Ontdek de ZX-Spectrum van Tim Hartnell
Flitsend FORTH van Alan Winfield

PROGRAMMEREN/PROGRAMMEERTALEN

Een methode van programmeren van prof.dr. Edsger W. Dijkstra en ir. W.H.J. Feijen
Programmeren, het ontwerpen van algoritmen, in Pascal van ir. J.J. van Amstel
Inleiding tot het programmeren, deel 1 van ir. J.J. van Amstel e.a.
Inleiding tot het programmeren, deel 2 van ir. J.J. van Amstel e.a.
JSP - Jackson struktureel programmeren van Henk Jansen
Aspecten van programmeertalen van ir. J.J. van Amstel en ir. J.A.A.M. Poirters
Programmeertalen, een inleiding van ir. J.J. van Amstel e.a.
Het Groot Pascal Spreukenboek van H.F. Ledgard, P.A. Nagin en J.F. Hueras
Cursus Pascal van prof.dr. A. van der Sluis en drs. C.A.C. Görts
Cursus eenvoudig Pascal van prof.dr. A. van der Sluis en drs. C.A.C. Görts
Inleiding programmeren in Pascal van C. van de Wijgaart
BASIC, EIT-serie, deel 3
Cursus BASIC van ir. R. Bloothoofd e.a.
Cursus COBOL van dr. A. Parkin
Struktuur en stijl in COBOL van ir. E. Dürr en dr.ir. F. Mulder
Cursus FORTRAN 77 van J.N.P. Hume en R.C. Holt
Programmeren in LISP van prof.dr. L.L. Steels
Cursus ALGOL 60 van prof.dr. A. van der Sluis en drs. C.A.C. Görts
Programmeren, deel 2: Van analyse tot algoritme van prof.drs. C. Bron
Inleiding programmeren en programmeertechnieken, EIT-serie, deel 1
Inleiding programmeren van prof.dr. R.J. Lunbeck

SYSTEEMPROGRAMMATUUR

Bedrijfssystemen, EIT-serie, deel 4

Systeemprogrammatuur van drs. H. Alblas

Vertalerbouw van drs. H. Alblas e.a.

Unix, het standaard operating system van G.J.M. Austen en H.J. Thomassen

BESTANDSORGANISATIE/DATABASES

Informatiestructuren, bestandsorganisatie en bestandsontwerp, EIT-serie, deel 5

Gegevensstructuren van R. Engmann e.a.

Bestandsorganisatie van prof.dr. R.J. Lunbeck en drs. F. Remmen

Databases van drs. F. Remmen

INFORMATIEANALYSE/SYSTEEMONTWERP

Effectieve toepassingen van computers van M. Peltu

Voorbereiding van computertoepassingen van prof.dr. A.B. Frielink

Simulatie, een moderne methode van onderzoek van drs. S.K.T. Boersma en ir. T. Hoenderkamp

Systeemontwikkeling volgens SDM van H.B. Eilers

Een samenvatting van de System Development Methodology SDM van PANDATA

Cases op het gebied van administratieve organisatie en informatieverzorging (inclusief systeemontwerp) van prof.dr. P.G. Bosch en H.A. te Rijdt

Uitwerkingenboek bij Cases van prof.dr. P.G. Bosch en H.A. te Rijdt

Gegevensanalyse van R.P. Langerhorst

Evaluation of methods and techniques for the analysis, design and implementation of information systems, editors: J. Blank en M.J. Krijger

Analyse van de informatiebehoeften en de inhoudsbeschrijving van een databank van prof.dr. P.G. Bosch en ir. H.M. Heemskerk

Eerlijk en helder van prof.dr. P.G. Bosch

THEORETISCH/COMPUTERSCHAAK/TOEPASSINGEN

Computers in de negentiger jaren van G.L. Simons

Expert systemen, ontwikkeling en gebruik van intelligente programma's van Henk de Swaan Arons en Peter van Lith

Abstracte automaten en grammatica's van prof.dr. A. Ollongren en ir. Th.P. van der Weide

De tekstmachine van dr. M. Boot en drs. H. Koppelaar

Computerschaak, schaakwereld en kunstmatige intelligentie van dr. H.J. van den Herik

Lineaire programmering als hulpmiddel bij de besluitvorming van prof.dr. S.W. Douma

Simulatie en sociale systemen, redaktie: J.L.A. Geurts en J.H.L. Oud

Onderneming en overheid in systeem-dynamisch perspectief, redaktie: A.F.G. Hanken en J.H.L. Oud

Over de inhoud van dit boek

Dit is eigenlijk een boek voor elke commodore 64-gebruiker. Het boek bevat 40 (re)creatieve en (ont)spannende grafische programma's, voorzien van tekst en uitleg.

Diegenen die de programma's gewoon intikken en draaien zullen zelfs zonder die uitleg verrukt zijn over hetgeen de programma's laten zien.

De theorie achter de programma's berust veelal op wiskundige begrippen. De lezer, die iets van de sinus en cosinus afweet zal hiermee geen moeite hebben. Er zijn o.a. programma's voor het drie-dimensionaal tekenen, voor LOGO-graphics en programma's met grafische toepassingen.

Leraren en leerlingen kunnen met menig programma hun voordeel doen.

Het boek bevat ook een BASIC-programma voor het inlezen van acht machinetaalroutines, waarmee gebruikers van de commodore 64, die niet over Simon's BASIC beschikken, toch op een eenvoudige manier met hoge resolutie graphics kunnen werken. Dit zijn o.a. routines voor het in- en uitschakelen van de hoge resolutiestand, voor het tekenen van punten en lijnen op het hoge resolutiescherm en voor het kleuren van vlakken.

Marcel Sutter

